QUERY COMPLEXITY OF ELICITING CONSENSUS LOTTERIES WITH BOUNDED ADAPTIVITY AND PARALLELISM

Anonymous authors
Paper under double-blind review

ABSTRACT

When a decision requires unanimous approval, limited opportunities for feedback can make elicitation costly. We study how to find a consensus lottery (i.e., a distribution over alternatives accepted by every agent) or determine that none exists, using only binary accept/reject queries. Each agent accepts a lottery when its expected utility meets an unknown threshold. We consider two interaction constraints: *bounded adaptivity*, which limits the number of feedback rounds, and *bounded parallelism*, which limits the total number of queries per round. For arbitrary numbers of agents and alternatives, we give algorithms and lower bounds that quantify the costs of learning acceptance constraints and finding consensus under these limits. We determine the optimal round and query complexities for learning one agent's acceptable set under bounded parallelism, and develop consensus algorithms that learn all constraints in parallel, as well as algorithms that find consensus without fully learning every agent's constraint. For two alternatives, we obtain tight bounds in several settings and demonstrate that randomization can substantially reduce the number of queries needed within a fixed number of rounds and the number of rounds needed under bounded parallelism, even though it cannot improve the optimal total query count when rounds are unrestricted.

1 INTRODUCTION

Many ML systems are deployed only after several stakeholders confirm that a proposed deployment plan is acceptable. A safety team may impose risk limits, operators may require reliability guarantees, legal or policy reviewers may impose compliance constraints, and domain experts may rule out clinically or operationally unsafe behavior (International Organization for Standardization, 2018; National Institute of Standards and Technology, 2018; AI Now Institute et al., 2020; Raji et al., 2020; Busuioc, 2021; National Institute of Standards and Technology, 2023; Labkoff et al., 2024). The goal is then to find a plan acceptable to everyone, rather than optimize a single aggregate score.

In many of these settings, the plan may randomize between model variants (Kohavi et al., 2020; Tang et al., 2010), mix conservative and aggressive operating modes (Altman, 1999; García & Fernández, 2015), or use a fallback policy with specified probability (Geifman & El-Yaniv, 2017). We call a distribution over a finite menu of alternatives a *lottery*, and one accepted by every stakeholder a *consensus lottery*.

A mixture can be acceptable even when neither pure alternative is. Suppose the menu contains a conservative fallback s_1 and a higher-performing model s_2. A safety reviewer accepts plans that use s_2 at most 30% of the time, while an operations reviewer requires it to be used at least 20% of the time. Neither model alone is acceptable to both reviewers, but any mixture using s_2 between 20% and 30% of the time is. The challenge, however, is informational. Stakeholders may not be able or willing to report complete utilities over all mixtures, but they can often say whether a concrete proposal is acceptable to them. This motivates a *membership query* model, where the algorithm proposes a mixture to a reviewer and observes an accept/reject response. We model each agent's acceptable set as an unknown halfspace in the probability simplex. Choo et al. (2026) study consensus elicitation with fully adaptive accept/reject queries and show that it need not require learning every agent's constraint.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

In this paper, we ask what remains possible when *interaction* is limited. A limit of R rounds restricts how often later queries can use earlier answers. A limit of B queries per round (or a *batch cap*) instead restricts how many proposals can be evaluated before the next round. These are different resources: querying n agents at the same lottery can take one round with unrestricted batches, but requires $\lceil n/B \rceil$ rounds under a batch cap. Review meetings may limit the number of rounds, whereas evaluator capacity may limit the batch size. Our central question is how limits on interaction affect the number of queries and rounds needed to find a consensus lottery or determine that none exists, and when this can be done without learning every agent's constraint.

1.1 Our Results

We study how limited interaction affects both learning agents' acceptance constraints and finding consensus. Here n is the number of agents, m is the number of alternatives, and ε is the parameter precision, with ε-quantized parameters as in Section 2. Building on the fully adaptive analysis of Choo et al. (2026), we give bounds for arbitrary m and stronger guarantees for two alternatives.

Arbitrary numbers of alternatives. Our fixed-round algorithms for learning one agent's acceptable set must choose which edges of the probability simplex to query as well as find their thresholds. Our deterministic algorithms use $\mathcal{O}(m^2 + Rm^{1+1/R}\varepsilon^{-2/R})$ queries for $R \geq 2$ and $\mathcal{O}(mR\varepsilon^{-2/(R-2)})$ queries for $R \geq 3$ (Theorems 3.1–3.2). The first begins searching before it knows which edges are needed; the second uses two rounds to select the edges first. Neither bound is uniformly smaller. For learning one acceptable set, Theorem 3.3 gives the randomized lower bound $\Omega(R\varepsilon^{-2/R} + Rm\varepsilon^{-1/R})$ for fixed R and sufficiently small ε. Learning all n agents in parallel then gives consensus algorithms using the same number of rounds and n times the queries (Theorem 3.4).

Under a batch cap, we determine the optimal round and total query complexities for learning one agent's acceptable set (Theorem 4.2). For consensus, we give a lower bound that also accounts for checking agents that do not affect the consensus lottery, an algorithm for learning selected agents together, and a batch-limited implementation of the weighted-sampling method of Choo et al. (2026) (Theorems 4.3–4.5). The sampling method learns selected constraints and checks all agents at the resulting candidate; its expected query bound has no factor n multiplying $\log(1/\varepsilon)$.

Two alternatives: finding consensus without learning every constraint. For $m = 2$, the optimal total query count is $\Theta(n + \log(1/\varepsilon))$ for every batch cap B, with no restriction on rounds (Theorem 4.6). A deterministic algorithm attains this bound by searching directly for a consensus lottery and removing agents known to accept every remaining candidate. This matches the total query lower bound of Choo et al. (2026, Theorem 4.1), closing the gap between their upper and lower bounds for two alternatives.

Two alternatives: randomization helps when interaction is limited. For $m = 2$ and every fixed R, the optimal deterministic R-round query count is $\Theta_R(n\varepsilon^{-2/R})$. An always-correct randomized algorithm uses at most three rounds and $\mathcal{O}(n + \sqrt{n}\,\varepsilon^{-2})$ worst-case expected queries (Theorem 3.5). Thus, when $n = \varepsilon^{-4}$, the optimal three-round query count is $\Theta(n^{7/6})$ deterministically but $\Theta(n)$ in expectation with randomization. Under a batch cap, we also determine the optimal deterministic and randomized round counts; the randomized expected count is $\Theta(1 + n/B + \log(1/\varepsilon)/\log(B+1))$ (Theorem 4.6). Table 1 illustrates how randomization can reduce the cost of limited interaction even though it cannot improve the optimal total query count when rounds are unrestricted.

Table 1: Two alternatives with $n = \varepsilon^{-4}$ as $\varepsilon \to 0$. All entries are optimal. Randomized costs are worst-case expectations, and their lower bounds allow error $1/3$.

Cost and interaction limit	Deterministic	Randomized
Total queries; rounds unrestricted	$\Theta(n)$	$\Theta(n)$
Total queries; at most three rounds	$\Theta(n^{7/6})$	$\Theta(n)$
Rounds; at most $B = n$ queries per round	$\Theta\left(\frac{\log(1/\varepsilon)}{\log\log(1/\varepsilon)}\right)$	$\Theta(1)$

The optimal multi-agent bounds for $m > 2$, and the general randomized fixed-round complexity even for $m = 2$, remain open.

1.2 Related Work

In this section, we discuss the work most directly related to our model. We refer the reader to Appendix A for further related work.

Consensus lotteries from membership queries. Choo et al. (2026) introduce the oracle model and study fully adaptive elicitation. We build on their reconstruction of an acceptable set from edge thresholds, their $\mathcal{O}(m \log(1/\varepsilon))$ halfspace learning algorithm, and their weighted-sampling method for consensus. Their total query bounds do not determine the number of queries needed within a fixed number of rounds or the optimal round count under a batch cap. Our two-alternative results determine both deterministic complexities and show that randomization can improve them, even though it cannot improve the optimal total query count. These improvements require more than batching each agent's threshold search: our direct search removes both candidate values and agents, and our sampling algorithms check acceptance of an entire candidate interval. Our matching total query lower bound holds even when every agent is nontrivial, and for randomized algorithms with error at most $1/3$.

Membership query learning and active learning of halfspaces. Our model connects to query-based concept learning (Angluin, 1988), active learning (Cohn et al., 1994; Settles, 2009; Dasgupta, 2005; Hanneke, 2014), generalized binary search (Nowak, 2011), and halfspace learning with membership queries (Kelner, 2020; Diakonikolas et al., 2024). Unlike PAC or distributional learning, our goal is to return an exact consensus lottery or correctly report that none exists, for parameters with finite precision. Exact learning of geometric sets has also been studied (Bshouty et al., 1998; Goldberg & Kwek, 2000); we discuss the different query models in Appendix A. Moreover, the hidden constraints belong to different agents and must be intersected: a bound for learning one halfspace does not account for checking all agents under a batch cap.

2 Preliminaries

For any integer $z \geq 1$, let $[z] := \{1, \ldots, z\}$. Let $N = [n]$ be the set of $n \geq 2$ agents and let $S = \{s_1, \ldots, s_m\}$ be the set of $m \geq 2$ alternatives. A lottery over S is a probability vector $\mathbf{x} = (x_1, \ldots, x_m) \in \Delta(S)$, where $\Delta(S) := \{\mathbf{x} \in \mathbb{R}^m_{\geq 0} : \sum_{j=1}^m x_j = 1\}$. For $j \in [m]$, let $\mathbf{e}_j$ denote the pure lottery that assigns probability 1 to s_j. Each agent $i \in N$ has an unknown utility vector $u_i = (u_i(s_1), \ldots, u_i(s_m)) \in [0,1]^m$ and an unknown acceptance threshold $\tau_i \in (0,1]$.[1] For a lottery $\mathbf{x} \in \Delta(S)$, the expected utility of agent i is $\langle u_i, \mathbf{x} \rangle = \sum_{j=1}^m x_j u_i(s_j)$. Agent i accepts $\mathbf{x}$ if and only if $\langle u_i, \mathbf{x} \rangle \geq \tau_i$. Thus its acceptable set is $A_i := \{\mathbf{x} \in \Delta(S) : \langle u_i, \mathbf{x} \rangle \geq \tau_i\}$. This is the intersection of the simplex with a closed halfspace. We exclude trivial agents by assuming $A_i \neq \varnothing$ and $A_i \neq \Delta(S)$ for every $i \in N$. Equivalently, each agent accepts at least one pure lottery and rejects at least one pure lottery.

The expected utility representation (von Neumann & Morgenstern, 1944) specifies a linear acceptance rule; agents need not report utilities or rankings. Linear reward models give a similar interpretation (Ng & Russell, 2000; Abbeel & Ng, 2004; Sadigh et al., 2017; Christiano et al., 2017). Appendix B.1 discusses the assumption and extensions.

Query oracle model. The algorithm does not observe u_i or τ_i directly. A query is a pair $(i, \mathbf{x}) \in N \times \Delta(S)$, and the oracle returns $\texttt{Query}(i, \mathbf{x}) := \texttt{True}$ if and only if $\mathbf{x} \in A_i$, and $\texttt{Query}(i, \mathbf{x}) := \texttt{False}$ otherwise.[2] The *(total) query complexity* of an execution is the number of oracle calls.

Consensus lotteries. Given query access to $(A_i)_{i \in N}$, the goal is to output either (i) a *consensus lottery* $\mathbf{x} \in \Delta(S)$ such that $\mathbf{x} \in \bigcap_{i \in N} A_i$, or (ii) NULL, in which case it must hold that $\bigcap_{i \in N} A_i = \varnothing$. A deterministic algorithm must terminate with a correct answer on every instance. A randomized algorithm may use internal randomness. It is *Las Vegas* if it terminates with probability one and returns a correct answer whenever it terminates, and it has *error at most* $1/3$ if it is correct with

[1] Agents with threshold 0 may be ignored, so assuming $\tau_i > 0$ is without loss of generality.

[2] When answers are written as vectors, we identify `True` with 1 and `False` with 0.

probability at least $2/3$ on every instance. Randomized costs are measured in worst-case expectation over the internal randomness. Any stated round or batch-size cap must hold on every execution. Our randomized upper bounds are Las Vegas, while several lower bounds hold even for algorithms with error at most $1/3$. Randomizing the algorithm's queries is distinct from returning a lottery: deterministic algorithms also return lotteries.

Rounds and batch size. An execution proceeds in rounds. In round r, the algorithm chooses a batch $Q_r = ((i_{r,1}, \mathbf{x}_{r,1}), \ldots, (i_{r,q_r}, \mathbf{x}_{r,q_r}))$, which may depend on the answers from earlier rounds and on the internal randomness, but not on answers from round r. The oracle then returns the answer vector $(\texttt{Query}(i_{r,1}, \mathbf{x}_{r,1}), \ldots, \texttt{Query}(i_{r,q_r}, \mathbf{x}_{r,q_r})) \in \{0,1\}^{q_r}$.

The *round complexity* is the worst-case number of rounds for a deterministic algorithm and the worst-case expected number for a randomized algorithm. The *batch size* is the largest number of queries in any round. An algorithm is R-round if it uses at most R rounds on every execution, and B-batched if $|Q_r| \le B$ in every round. The batch cap is global: B counts agent-lottery pairs across all agents, not queries per agent. An R-round, B-batched algorithm therefore uses at most RB queries on every execution.

Finite precision. Following Choo et al. (2026), fix $\varepsilon \in (0, 1/2]$ with $1/\varepsilon \in \mathbb{Z}_{>0}$. We assume ε-quantized parameters: $u_i(s_j) \in \{0, \varepsilon, 2\varepsilon, \ldots, 1\}$ and $\tau_i \in \{\varepsilon, 2\varepsilon, \ldots, 1\}$. The guarantees in this paper are exact on these quantized instances. Appendix B.2 shows that no deterministic algorithm using finitely many queries on every instance can solve the unquantized problem exactly, even for two agents and two alternatives.

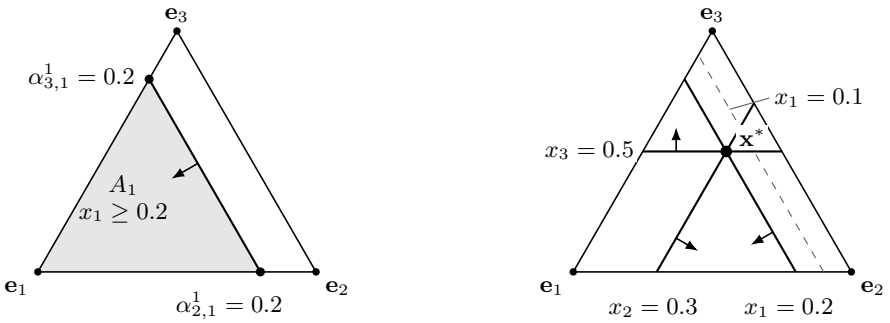
Edge thresholds. For distinct $k, j \in [m]$ and $\alpha \in [0,1]$, define $\mathbf{x}_{k,j,\alpha} := (1-\alpha)\mathbf{e}_k + \alpha \mathbf{e}_j$, the lottery on the oriented edge from $\mathbf{e}_k$ to $\mathbf{e}_j$. Fix an agent i that rejects $\mathbf{e}_k$ and accepts $\mathbf{e}_j$. Since $u_i(s_k) < \tau_i \le u_i(s_j)$, its response changes exactly once along this edge. The *edge threshold* $\alpha^i_{k,j} \in (0,1]$ is characterized by $\texttt{Query}(i, \mathbf{x}_{k,j,\alpha}) = \mathbf{1}[\alpha \ge \alpha^i_{k,j}]$. We write $\alpha_{k,j}$ when the agent is clear. This threshold is a scalar; the corresponding boundary point is $\mathbf{x}_{k,j,\alpha^i_{k,j}}$. If $\alpha^i_{k,j} = 1$, only the endpoint $\mathbf{e}_j$ is accepted. Many arguments below reduce halfspace learning to identifying a few such thresholds.

With ε-quantization, every edge threshold lies in the finite set $\Gamma_\varepsilon := \{\gamma \in (0,1] : \gamma = p/q \text{ in lowest terms for some } 1 \le q \le 1/\varepsilon\}$. Indeed, $\alpha^i_{k,j} = \frac{\tau_i - u_i(s_k)}{u_i(s_j) - u_i(s_k)}$: both differences are positive multiples of ε, and the denominator is at most 1, so the reduced denominator is at most $1/\varepsilon$. Thus exact edge threshold search is a search over Γ_ε, not merely over multiples of ε. For example, with $\varepsilon = 1/10$, utilities $u_i(s_k) = 0$, $u_i(s_j) = 0.3$ and threshold $\tau_i = 0.2$ give $\alpha^i_{k,j} = 2/3$. Quantization restricts the hidden parameters, not the lotteries that may be queried or returned. The denominator bound follows Choo et al. (2026, Appendix C.1); exact search for a rational with bounded denominator also appears in Goldberg & Kwek (2000, Lemma 3). The classical count of reduced fractions gives $|\Gamma_\varepsilon| = \Theta(\varepsilon^{-2})$ (Hardy & Wright, 2008, Theorem 330). Also let $\Gamma^-_\varepsilon := \Gamma_\varepsilon \setminus \{1\}$.

Running example. Let $m = 3$, $n = 5$, and $\varepsilon = 1/10$. Consider the following instance; the algorithm does not know the utilities or thresholds.

i	$u_i(s_1)$	$u_i(s_2)$	$u_i(s_3)$	τ_i
1	1	0	0	0.2
2	0	1	0	0.3
3	0	0	1	0.5
4, 5	1	0	0	0.1

The first three agents require $x_1 \ge 0.2$, $x_2 \ge 0.3$, and $x_3 \ge 0.5$. Since these lower bounds sum to one, their only common acceptable lottery is $\mathbf{x}^* = (0.2, 0.3, 0.5)$. Agents 4 and 5 accept it too, so $\mathbf{x}^*$ is the unique consensus lottery, although no pure lottery is accepted by everyone (Figure 1). Once agents 1–3 have been learned, the algorithm knows the only possible consensus lottery, but it still needs to establish that agents 4 and 5 accept it. Querying both at $\mathbf{x}^*$ takes one unrestricted round, or two rounds when $B = 1$. Appendix B.3 shows that this check cannot be skipped.



(a) Learning agent 1's constraint (b) Finding and checking a candidate

Figure 1: The running example. (a) The two marked boundary points determine agent 1's boundary; the shaded side is accepted. (b) The first three constraints have the unique common lottery $\mathbf{x}^* = (0.2, 0.3, 0.5)$. Arrows point toward the accepted side of each solid boundary. The dashed boundary belongs to agents 4 and 5, whose acceptance of $\mathbf{x}^*$ still has to be established.

3 Bounded Adaptivity (Number of Rounds)

We first consider the setting where the elicitation procedure is limited to R rounds, with no cap on queries per round. We study how many queries are needed under this limit, starting with one agent and then asking when consensus requires learning every agent's constraint.

3.1 Halfspace Learning for a Single Agent

For a fixed agent $i \in N$, *halfspace learning* means returning a vector $\mathbf{c}_i \in \mathbb{R}^m$ such that $A_i = \{\mathbf{x} \in \Delta(S) : \langle \mathbf{c}_i, \mathbf{x} \rangle \geq 1\}$. This is stronger than finding one accepted lottery: the learned inequality can be checked at any later candidate without another query to the agent. The fully adaptive algorithm of Choo et al. (2026) learns it from at most $m - 1$ edge thresholds. In Figure 1(a), two boundary points determine agent 1's boundary.

With only R rounds, choosing the edges is part of the challenge. Before receiving any answers, the algorithm does not know which pure lotteries are accepted or which edges are needed. The algorithm in Theorem 3.1 begins with a small number of queries on every ordered edge, including tests that distinguish $u_i(s_j) > \tau_i$ from $u_i(s_j) = \tau_i$. After the first answers arrive, it continues searching on at most $m - 1$ selected edges rather than paying the full search cost on every edge.

Theorem 3.1. *Fix an ε-quantized agent i. For $R = 1$, there is a deterministic one-round algorithm using $\mathcal{O}(m^2\varepsilon^{-2})$ queries. For every integer $R \geq 2$, there is a deterministic R-round algorithm with batch size $\mathcal{O}(m^2 + m^{1+1/R}\varepsilon^{-2/R})$ and total query complexity $\mathcal{O}(m^2 + Rm^{1+1/R}\varepsilon^{-2/R})$.*

The first round considers quadratically many edges, so we use fewer queries per edge than in later rounds. Balancing the batch sizes gives the factor $m^{1+1/R}$; the additive m^2 term covers initial tests. Next, we show that using two rounds to select the edges first can remove the factor $m^{1/R}$, but leaves only $R - 2$ rounds for exact searches. This improves the dependence on m while worsening that on ε.

Theorem 3.2. *Fix an ε-quantized agent i and an integer $R \geq 3$. There is a deterministic R-round algorithm with batch size $\mathcal{O}(m\varepsilon^{-2/(R-2)})$ and total query complexity $\mathcal{O}(mR\varepsilon^{-2/(R-2)})$.*

Note that neither upper bound is uniformly smaller. The next lower bound combines the cost of locating one edge threshold with a term that grows with the simplex dimension.

Theorem 3.3. *Fix $R \geq 1$ and assume ε-quantized parameters. For all sufficiently small ε, every R-round algorithm that learns one hidden halfspace exactly has worst-case query complexity $\Omega(R\varepsilon^{-2/R} + Rm\varepsilon^{-1/R})$. The bound holds for deterministic algorithms and for randomized algorithms with error at most $1/3$, using worst-case expected queries.*

3.2 MULTI-AGENT ALGORITHMS

We now consider the multi-agent problem of finding a consensus lottery, if one exists. We use the results of the previous section for single-agent halfspace learning. To combine learned inequalities, we use a deterministic rule: for a finite set $C \subseteq \mathbb{R}^m$, let $\texttt{Select}(C)$ return the lexicographically maximum lottery satisfying $\langle \mathbf{c}, \mathbf{x} \rangle \geq 1$ for every $\mathbf{c} \in C$, or NULL if none exists.[3] We count only queries, not this offline computation. Let $\texttt{LearnHalfspace}(i)$ denote a deterministic procedure that queries only agent i and returns its exact vector $\mathbf{c}_i$, as defined earlier.

Learning all agents in parallel. With unrestricted batches, learning all agents takes no more rounds than learning one, as we show with the following result.

Theorem 3.4. *Fix $R \geq 1$. Suppose that, for every $i \in N$, $\texttt{LearnHalfspace}(i)$ is an R-round algorithm that uses at most Q' queries in total and at most B' queries in any round. Run $\texttt{LearnHalfspace}(i)$ for all $i \in N$ in parallel: in global round r, issue the round-r batch of every procedure that has not yet finished. After obtaining $(\mathbf{c}_i)_{i \in N}$, return $\texttt{Select}(\{\mathbf{c}_i : i \in N\})$. This procedure outputs a consensus lottery when one exists and* NULL *otherwise. Moreover, it is R-round, has batch size at most nB', and uses at most nQ' queries in total.*

Thus, either single-agent bound gives a consensus algorithm in the same R rounds, with its query count and batch size multiplied by n. In Appendix C.5, we give an alternative that learns only agents rejecting the candidate selected from the learned constraints. It can save queries on some instances but may exceed R rounds.

3.3 TWO ALTERNATIVES: THE BENEFIT OF RANDOMIZATION

We write every lottery as $\mathbf{x}_{1,2,\beta}$ (as in the preliminaries), where β is the probability assigned to s_2. By nontriviality, each agent accepts exactly one pure lottery. A *lower-bound agent* i rejects $\mathbf{e}_1$ and accepts precisely when $\beta \geq \alpha^i_{1,2}$; an *upper-bound agent* i accepts $\mathbf{e}_1$ and accepts precisely when $\beta \leq 1 - \alpha^i_{2,1}$. Thus, the consensus problem asks whether the largest lower bound is at most the smallest upper bound. The challenge is that both these values and the agents imposing them are unknown.

Theorem 3.5. *Assume $m = 2$ and ε-quantized parameters. For every fixed integer $R \geq 1$, the optimal deterministic R-round query complexity of consensus elicitation is $\Theta_R(n\varepsilon^{-2/R})$, where the constants may depend on R. There is also a randomized algorithm using at most three rounds on every execution and $\mathcal{O}(n + \sqrt{n}\,\varepsilon^{-2})$ worst-case expected queries.*

The deterministic lower bound must account for both the possible consensus values and the agents imposing them. Its proof maintains a set of candidate values and many agents whose thresholds may still be any of these values. With few rounds, the algorithm must query many of these agents in enough detail to distinguish different consensus values.

The randomized algorithm combines the sampling estimate of Gärtner & Welzl (2001) with endpoint tests and exact threshold learning in three rounds. First, query every agent at $\mathbf{e}_1$ to learn its inequality direction. In the same batch, fully learn a uniform sample of $\lceil \sqrt{n} \rceil$ agents by querying each at all values in $\{0\} \cup \Gamma_\varepsilon$. If their acceptable intervals have empty intersection, return NULL; otherwise, the intersection contains every possible consensus value. Second, query each remaining agent at the endpoint most difficult for it to accept: the left endpoint for a lower-bound agent, and the right endpoint for an upper-bound agent. An agent that accepts this test accepts *every value in the interval* and needs no further queries. Third, fully learn only the agents that rejected these tests and intersect their constraints with the sampled interval. No final checking round is needed.

Figure 2 shows why the endpoint choice matters. A lower-bound agent requiring $\beta \geq 0.1$ accepts the test at 0.2 and is removed without learning its threshold; an agent requiring $\beta \geq 0.4$ rejects and needs further queries. In general, an agent remains only if its bound is stricter than every sampled bound of its direction. A uniform sample of $\lceil \sqrt{n} \rceil$ agents leaves $\mathcal{O}(\sqrt{n})$ such agents in expectation. Thus the algorithm pays for $\mathcal{O}(n)$ simple checks and only $\mathcal{O}(\sqrt{n})$ full endpoint searches in expectation (see Appendix C.8 for more details).

[3] This can be implemented by linear programs that maximize x_1, then x_2, and so on, over the feasible set.

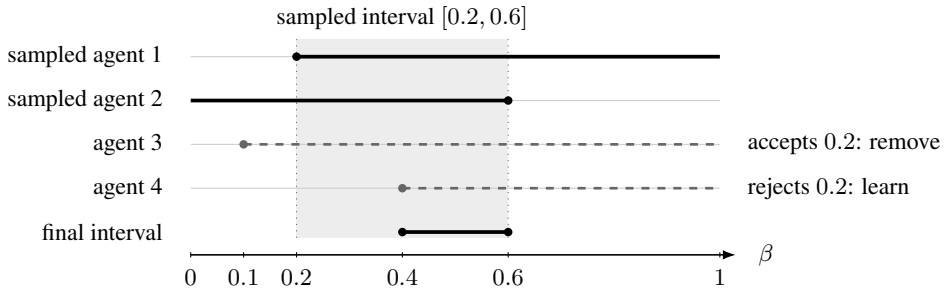


Figure 2: Endpoint checks in the three-round algorithm ($n = 4$, $\varepsilon = 1/10$). The sample restricts consensus to $[0.2, 0.6]$. Agent 3 accepts the left endpoint and therefore every value in this interval. Agent 4 rejects it and is learned in round 3, leaving $[0.4, 0.6]$. Dashed intervals illustrate the unsampled agents' hidden acceptable sets; the algorithm does not know them before the tests.

We note, however, that randomization cannot give the same saving in one round: the optimal query count is $\Theta(n\varepsilon^{-2})$, even with error at most $1/3$ (Theorem C.5). For every fixed R, randomized R-round algorithms still need $\Omega(n + R\varepsilon^{-2/R})$ queries (Theorem C.6). These bounds leave the exact randomized fixed-round complexity open for general n and ε.

4 Bounded Parallelism (Number of Queries Per Round)

We now consider the setting where each round of the elicitation procedure is limited to B queries in total across all agents. The algorithm must divide this capacity between learning constraints and checking candidates. We first analyze threshold search on one edge, then combine edge searches for one or several agents. Finally, for two alternatives, we show how to find consensus without learning every constraint.

4.1 Threshold Search (Single Edge)

We begin with the edge searches used to learn an agent's acceptance constraint. On an edge from a rejected pure lottery $\mathbf{e}_k$ to an accepted pure lottery $\mathbf{e}_j$, identifying $\alpha^i_{k,j}$ amounts to searching the ordered set Γ_ε. The following lemma describes how the batch cap limits each round: B queries divide the possible threshold values into at most $B + 1$ consecutive groups. This is the standard method of ordered search with simultaneous queries (Brânzei et al., 2020). We give the exact round count for our binary threshold oracle.

Lemma 4.1. *Let $\Lambda = \{\lambda_1 < \cdots < \lambda_M\}$ be a finite ordered set, and suppose there is an unknown $\lambda^* \in \Lambda$. A query at $\lambda \in \Lambda$ returns $\mathbf{1}[\lambda \geq \lambda^*]$. Then, for every integer $B \geq 1$, the optimal deterministic B-batched round complexity for identifying λ^* exactly is $\lceil \log_{B+1} M \rceil$. Moreover, this is achievable using at most $B\lceil \log_{B+1} M \rceil$ queries in total.*

For example, $\Gamma_{1/4} = \{1/4, 1/3, 1/2, 2/3, 3/4, 1\}$. With $B = 2$, query the edge from $\mathbf{e}_k$ to $\mathbf{e}_j$ at $1/3$ and $2/3$. The answers $(1, 1)$ leave $\{1/4, 1/3\}$, $(0, 1)$ leave $\{1/2, 2/3\}$, and $(0, 0)$ leave $\{3/4, 1\}$. One more query identifies the threshold. Two simultaneous queries thus have three possible answer vectors, not four, since $(1, 0)$ cannot occur. For an edge threshold $\alpha^i_{k,j} \in \Gamma_\varepsilon$, Lemma 4.1 shows that the optimal deterministic number of rounds for finding it is exactly $\lceil \log_{B+1} |\Gamma_\varepsilon| \rceil$, since every candidate $\gamma = p/q \in \Gamma_\varepsilon$ can occur: take $u_i(s_k) = 0$, $u_i(s_j) = q\varepsilon$, and $\tau_i = p\varepsilon$.

4.2 Single Agent

We now use these edge searches to learn one agent's entire acceptable set, as defined in Section 3.1. The batch cap must cover both the initial queries used to choose the edges and the subsequent searches on at most $m - 1$ edges. Theorem 4.2 gives the optimal round and total query complexities. Its upper bound divides each batch among the edge searches; its lower bound applies to arbitrary lottery queries, so querying elsewhere in the simplex cannot improve the asymptotic bounds.

Theorem 4.2. *Fix an ε-quantized agent. The optimal deterministic B-batched round complexity for learning its acceptable set exactly is $\Theta(\frac{m}{B} \log \frac{1}{\varepsilon})$ for $1 \leq B \leq m - 1$, and $\Theta(1 + \frac{\log(1/\varepsilon)}{\log(1+B/m)})$*

for $B \geq m-1$. For every $B \geq 1$, the optimal total query complexity is $\Theta(m \log(1/\varepsilon))$. The lower bounds hold for randomized algorithms with error at most $1/3$, measured in worst-case expected rounds or queries.

When $B < m-1$, not all required edges can be queried in the same round, producing the factor m/B. When $B \geq m-1$, extra queries on each edge reduce the number of rounds needed to search it. Minimizing rounds and minimizing total queries need not use the same algorithm: binary search on each edge minimizes queries, whereas several queries per edge can save rounds.

4.3 Several agents

We now return to consensus elicitation. Learning every agent's acceptable set is sufficient, but an algorithm may use fewer queries by learning only some constraints and checking the other agents at a candidate lottery. We first quantify a cost that such checking cannot avoid, before giving algorithms for learning all agents or selected subsets under the same global cap B.

Theorem 4.3. *Assume $n \geq m$ and $m \leq (1/\varepsilon)^{1-\delta}$ for a fixed constant $\delta \in (0,1)$. For all sufficiently small ε, there is one family of valid instances such that every B-batched algorithm requires $\Omega(\frac{n-m}{B} + \frac{m}{B}\log\frac{1}{\varepsilon})$ rounds for $1 \leq B \leq m-1$, and $\Omega(\frac{n-m}{B} + 1 + \frac{\log(1/\varepsilon)}{\log(1+B/m)})$ rounds for $B \geq m-1$. The bound holds for deterministic algorithms and for randomized algorithms with error at most $1/3$, using worst-case expected rounds.*

The term $(n-m)/B$ accounts for checking agents that do not affect the consensus lottery. As in Figure 1(b), such agents accept the uniquely determined candidate, but their acceptance is unknown until they are queried. The other term is the cost of locating the required lottery. The condition on m is used in the counting argument for this second term.

Joint learning of a chosen set. To obtain an upper bound, we first learn the constraints of several agents together. Queries to all chosen agents count toward the same batch cap B. We obtain the following result.

Theorem 4.4. *Fix a set of $k \geq 1$ ε-quantized agents. For every $B \geq 1$, there is a deterministic B-batched algorithm that learns all k acceptable sets exactly in $\mathcal{O}(1 + \frac{\log(1/\varepsilon)}{\log(1+B/(km))})$ rounds. A possibly different deterministic B-batched algorithm uses $\mathcal{O}(km\log(1/\varepsilon))$ total queries.*

The algorithm proving the round bound first queries each chosen agent at all pure lotteries, then uses one additional query per accepted pure lottery to determine which edge thresholds must be found. It searches the resulting at most $k(m-1)$ edges in batches that may contain queries for different agents. For $k = n$, applying `Select` to the learned inequalities solves consensus.

Batch-limited weighted sampling. The weighted-sampling method of Choo et al. (2026) follows the linear programming algorithm of Clarkson (1995). It learns the constraints of a small sample of agents, computes a candidate from them, and queries all agents at that candidate. Agents rejecting earlier candidates receive greater sampling weight. Our next result bounds the round and query complexities of combining this method with Theorem 4.4, counting both sample learning and candidate checks under the same global cap B.

Theorem 4.5. *Let $k := \min\{n, \lceil 6m^2 \rceil\}$. Suppose that any specified set of at most k agents is learned jointly by a deterministic B-batched routine using at most R' rounds and Q' queries. There is a randomized B-batched consensus algorithm with worst-case expected complexity $\mathcal{O}(m\log(n+1)(\lceil \frac{n}{B} \rceil + R'))$ rounds and $\mathcal{O}(m\log(n+1)(n+Q'))$ queries. One may always take the smaller of these bounds and the direct all-agent bounds from Theorem 4.4.*

Using the round bound in Theorem 4.4 with $k = \mathcal{O}(m^2)$ gives an expected round count of $\mathcal{O}(m\log(n+1)[\lceil \frac{n}{B} \rceil + 1 + \frac{\log(1/\varepsilon)}{\log(1+B/m^3)}])$. Using its total query bound instead gives $\mathcal{O}(m\log(n+1)[n + m^3\log(1/\varepsilon)])$ expected queries. These bounds may be attained by different algorithms. In both, the term involving $\log(1/\varepsilon)$ is not multiplied by n (Appendix D.8).

Two alternatives: optimal rounds and total queries. For two alternatives, searching directly for a consensus value avoids separately identifying the largest lower bound and smallest upper bound, as we show with the following result.

Theorem 4.6. *Assume $m = 2$ and ε-quantized parameters. For every batch cap $B \geq 1$, the optimal randomized expected round complexity is $\Theta(1 + \frac{n}{B} + \frac{\log(1/\varepsilon)}{\log(B+1)})$. The randomized round lower bound holds even with error at most $1/3$. The optimal deterministic worst-case round complexity is $\Theta(1+\frac{n}{B}+\frac{\log(1/\varepsilon)}{\log(2+\min\{B,\, B\log(1/\varepsilon)/n\})})$. For every B, the optimal total query count is $\Theta(n+\log(1/\varepsilon))$. This query bound is achieved deterministically and is necessary even with error at most $1/3$. The algorithms minimizing rounds and total queries need not be the same.*

Every nonempty consensus interval contains a value in $\{0\} \cup \Gamma_\varepsilon$: its left endpoint is 0 or an agent's lower bound. Algorithm 1 searches these candidates directly.

Algorithm 1 Direct feasibility search for two alternatives (one query at a time)

```
Query every agent at e_1 to determine its inequality direction.
Keep all agents and the ordered candidate values {0} ∪ Γ_ε.
while candidate values remain do
    Choose a median candidate β.
    Query the remaining agents at x_{1,2,β}, alternating lower- and upper-bound agents,
     until the first rejection. If one direction is exhausted, continue with the other.
    if every remaining agent accepts then return x_{1,2,β}
    end if
    Keep candidates above β after a lower-bound rejection, and below β otherwise.
    Remove the agents of the rejecting agent's direction that accepted x_{1,2,β}
     in this iteration. Keep the rejecting agent and all other agents.
end while
return NULL
```

Suppose one lower-bound agent accepts $\mathbf{x}_{1,2,1/2}$ and a later one rejects it. All feasible values now have $\beta > 1/2$, so the earlier agent accepts every remaining candidate and needs no further queries. An upper-bound rejection is symmetric. Each rejection at a median at least halves the candidate set. We call the queries at one candidate a *scan*. By alternating directions, a scan ending in a rejection uses at most two queries per removed agent, plus two. Each agent is removed at most once; a final scan in which all remaining agents accept checks each of them once. Including the direction queries, this gives $\mathcal{O}(n + \log(1/\varepsilon))$, since $|\Gamma_\varepsilon| = \mathcal{O}(\varepsilon^{-2})$.

To attain the randomized round bound, perform the sampling and endpoint checks of Figure 2 twice, using up to $\lceil\sqrt{B}\rceil$ agents per sample. Only $\mathcal{O}(n/B)$ agents remain in expectation. A batched version of Algorithm 1 then attains the round bound in Theorem 4.6; Appendix D.9 gives the algorithms and matching lower bounds.

5 Conclusion

We studied exact consensus elicitation under limits on adaptive rounds and on queries per round. For two alternatives, direct feasibility search attains the optimal $\Theta(n + \log(1/\varepsilon))$ total query bound without learning every constraint. We determine the optimal deterministic fixed-round query counts and both deterministic and randomized batch-limited round counts. Randomization does not improve the total query optimum, but can substantially reduce the cost of limited interaction. For arbitrary numbers of alternatives, our algorithms and lower bounds quantify the costs of learning constraints and checking candidates.

Our model assumes ε-quantized parameters, a fixed halfspace per agent, and noiseless answers. Without a precision restriction, deterministic finite-query exact elicitation is impossible in general (Appendix B.2); human answers may also be inconsistent or nonlinear. The exact randomized fixed-round complexity remains open for general n and ε, even for $m = 2$. For $m > 2$, gaps remain under both interaction limits. Interesting further directions include considering approximate consensus, noisy answers, and richer acceptable sets.

AI USE STATEMENT

In this work, we used generative AI tools to polish the exposition of the paper, including both the prose and the presentation of technical proofs, with the aim of improving clarity and readability. We also used generative AI tools to help identify relevant existing literature that we may have overlooked. We did not use generative AI tools to develop or formulate our model or proofs. Other uses (including generating synthetic datasets; designing or providing feedback on experiments; cleaning or reformatting datasets; supporting qualitative or thematic data analysis; and interpreting results) are not applicable to this work. All AI-assisted revisions and literature suggestions were reviewed and independently verified by the authors. In particular, the mathematical arguments and proofs were checked by the authors for correctness, and literature identified with the assistance of generative AI was verified against the original sources for relevance and accuracy. The authors take full responsibility for the final content of this work, including any text, claims, or other material produced with the assistance of generative AI.

REFERENCES

Hasan Abasi, Ali Z. Abdi, and Nader H. Bshouty. Learning Boolean halfspaces with small weights from membership queries. *Theoretical Computer Science*, 650:45–56, 2016.

Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, 2004.

AI Now Institute, City of Amsterdam, City of Helsinki, Mozilla Foundation, and Nesta. Using procurement instruments to ensure trustworthy AI, 2020. Position paper.

Eitan Altman. *Constrained Markov Decision Processes*. Chapman and Hall, 1999.

Dana Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988.

Maria-Florina Balcan, Alina Beygelzimer, and John Langford. Agnostic active learning. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pp. 65–72, 2006.

Eric Balkanski and Yaron Singer. The adaptive complexity of maximizing a submodular function. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pp. 1138–1151, 2018.

Eric Balkanski, Aviad Rubinstein, and Yaron Singer. An optimal approximation for submodular maximization under a matroid constraint in the adaptive complexity model. *Operations Research*, 70(5):2967–2981, 2022.

Xiaohui Bei, Ning Chen, and Shengyu Zhang. Solving linear programming with constraints unknown. In *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming (ICALP)*, pp. 129–142, 2015.

Erdem Biyik, Nima Anari, and Dorsa Sadigh. Batch active learning of reward functions from human preferences. *ACM Transactions on Human-Robot Interaction*, 13(2):24:1–24:27, 2024.

Avrim Blum, Jeffrey Jackson, Tuomas Sandholm, and Martin Zinkevich. Preference elicitation and query learning. *Journal of Machine Learning Research*, 5:649–667, 2004.

Anna Bogomolnaia, Hervé Moulin, and Richard Stong. Collective choice under dichotomous preferences. *Journal of Economic Theory*, 122(2):165–184, 2005.

Craig Boutilier, Relu Patrascu, Pascal Poupart, and Dale Schuurmans. Constraint-based optimization and utility elicitation using the minimax decision criterion. *Artificial Intelligence*, 170(8–9): 686–713, 2006.

Florian Brandl, Felix Brandt, and Hans Georg Seedig. Consistent probabilistic social choice. *Econometrica*, 84(5):1839–1880, 2016.

Felix Brandt. Rolling the dice: Recent results in probabilistic social choice. In Ulle Endriss (ed.), *Trends in Computational Social Choice*, chapter 1, pp. 3–26. AI Access, 2017.

Simina Branzei and Jiawei Li. The query complexity of local search and brouwer in rounds. In *Proceedings of 35th Conference on Learning Theory (COLT)*, pp. 5128–5145, 2022.

Simina Brânzei and Noam Nisan. Communication complexity of cake cutting. In *Proceedings of the 20th ACM Conference on Economics and Computation (EC)*, pp. 525, 2019. Full version available as arXiv:1709.09876.

Simina Brânzei, Dimitris Paparas, and Nicholas J. Recker. Searching, sorting, and cake cutting in rounds. *arXiv preprint arXiv:2012.00738*, 2020.

Darius Braziunas and Craig Boutilier. Elicitation of factored utilities. *AI Magazine*, 29(4):79–92, 2008.

Nader H. Bshouty, Paul W. Goldberg, Sally A. Goldman, and H. David Mathias. Exact learning of discretized geometric concepts. *SIAM Journal on Computing*, 28(2):674–699, 1998.

R. C. Buck. Partition of space. *The American Mathematical Monthly*, 50(9):541–544, 1943.

Madalina Busuioc. Accountable artificial intelligence: Holding algorithms to account. *Public Administration Review*, 81(5):825–836, 2021.

Clément L. Canonne and Tom Gur. An adaptivity hierarchy theorem for property testing. *Computational Complexity*, 27:671–716, 2018.

Yuxin Chen and Andreas Krause. Near-optimal Batch Mode Active Learning and Adaptive Submodular Optimization. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pp. 160–168, 2013.

Davin Choo and Kirankumar Shiragur. Adaptivity complexity for causal graph discovery. In *Proceedings of the 39th Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 391–402, 2023.

Davin Choo, Paul W. Goldberg, and Nicholas Teh. Learning unanimously acceptable lotteries via queries. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026. Extended version available as arXiv:2604.17505.

Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 4302–4310, 2017.

Kenneth L. Clarkson. Las vegas algorithms for linear and integer programming when the dimension is small. *Journal of the ACM*, 42(2):488–499, 1995.

David A. Cohn, Les Atlas, and Richard E. Ladner. Improving generalization with active learning. *Machine Learning*, 15:201–221, 1994.

Sanjoy Dasgupta. Coarse sample complexity bounds for active learning. In *Proceedings of the 19th International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 235–242, 2005.

Ilias Diakonikolas, Daniel M. Kane, and Mingchen Ma. Active learning of general halfspaces: Label queries vs membership queries. In *Proceedings of the 38th International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 49180–49213, 2024.

Javier García and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

Bernd Gärtner and Emo Welzl. A simple sampling lemma: Analysis and applications in geometric optimization. *Discrete & Computational Geometry*, 25:569–590, 2001.

Bernd Gärtner, Jiří Matoušek, Leo Rüst, and Petr Škovroň. Violator spaces: Structure and algorithms. *Discrete Applied Mathematics*, 156(11):2124–2141, 2008.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 4885–4894, 2017.

Allan Gibbard. Manipulation of schemes that mix voting with chance. *Econometrica*, 45(3):665–681, 1977.

Paul W. Goldberg and Stephen Kwek. The precision of query points as a resource for learning convex polytopes with membership queries. In *Proceedings of the 13th Annual Conference on Computational Learning Theory (COLT)*, pp. 225–235, 2000.

Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

Yuhong Guo and Dale Schuurmans. Discriminative batch mode active learning. In *Proceedings of the 21st International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 593–600, 2007.

Steve Hanneke. Theory of disagreement-based active learning. *Foundations and Trends in Machine Learning*, 7(2–3):131–309, 2014.

G. H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers*. Oxford University Press, sixth edition, 2008.

Eduard Helly. Über mengen konvexer körper mit gemeinschaftlichen punkten. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 32:175–176, 1923.

Steven C. H. Hoi, Rong Jin, Jianke Zhu, and Michael R. Lyu. Batch mode active learning and its application to medical image classification. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pp. 417–424, 2006.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

International Organization for Standardization. ISO 31000:2018: Risk management—guidelines, 2018.

Ori Kelner. Learning halfspaces with membership queries. *arXiv preprint arXiv:2012.10985*, 2020.

Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. BatchBALD: Efficient and diverse batch acquisition for deep bayesian active learning. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 7026–7037, 2019.

Ron Kohavi, Diane Tang, and Ya Xu. *Trustworthy Online Controlled Experiments: A Practical Guide to A/B Testing*. Cambridge University Press, 2020.

Leon G. Kraft. A device for quantizing, grouping, and coding amplitude-modulated pulses. Master's thesis, Massachusetts Institute of Technology, 1949.

Steven Labkoff, Bilikis Oladimeji, Joseph Kannry, Anthony Solomonides, Russell Leftwich, Eileen Koski, Amanda L. Joseph, Monica Lopez-Gonzalez, Lee A. Fleisher, Kimberly Nolen, Sayon Dutta, Deborah R. Levy, Amy Price, Paul J. Barr, Jonathan D. Hron, Baihan Lin, Gyana Srivastava, Nuria Pastor, Unai Sanchez Luque, Tien Thi Thuy Bui, Reva Singh, Tayler Williams, Mark G. Weiner, Tristan Naumann, Dean F. Sittig, Gretchen Purcell Jackson, and Yuri Quintana. Toward a responsible future: Recommendations for AI-enabled clinical decision support. *Journal of the American Medical Informatics Association*, 31(11):2730–2739, 2024.

National Institute of Standards and Technology. Risk management framework for information systems and organizations: A system life cycle approach for security and privacy. Technical Report NIST SP 800-37 Rev. 2, National Institute of Standards and Technology, 2018.

National Institute of Standards and Technology. Artificial intelligence risk management framework (AI RMF 1.0). Technical Report NIST AI 100-1, National Institute of Standards and Technology, 2023.

Andrew Y. Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, pp. 663–670, 2000.

Robert D. Nowak. The geometry of generalized binary search. *IEEE Transactions on Information Theory*, 57(12):7893–7906, 2011.

Inioluwa Deborah Raji, Andrew Smart, Rebecca N. White, Margaret Mitchell, Timnit Gebru, Ben Hutchinson, Jamila Smith-Loud, Daniel Theron, and Parker Barnes. Closing the AI accountability gap: Defining an end-to-end framework for internal algorithmic auditing. In *Proceedings of the 3rd Conference on Fairness, Accountability, and Transparency (FAccT)*, pp. 33–44, 2020.

Dorsa Sadigh, Anca D. Dragan, Shankar Sastry, and Sanjit A. Seshia. Active preference-based learning of reward functions. In *Proceedings of the 13th Robotics: Science and Systems Conference (RSS)*, 2017.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

Diane Tang, Ashish Agarwal, Deirdre O'Brien, and Mike Meyer. Overlapping experiment infrastructure: More, better, faster experimentation. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 17–26, 2010.

Paolo Viappiani and Craig Boutilier. Optimal bayesian recommendation sets and myopically optimal choice query sets. In *Proceedings of the 24th International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 2352–2360, 2010.

John von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18:4945–4990, 2017.

Brian D. Ziebart, Andrew L. Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI)*, pp. 1433–1438, 2008.

A Further Related Work

Probabilistic social choice. A related strand of work is probabilistic social choice and collective choice under dichotomous preferences (Gibbard, 1977; Bogomolnaia et al., 2005; Brandl et al., 2016; Brandt, 2017). In that literature, agents' preferences or approval sets over deterministic alternatives are typically the input to a social choice rule, and the main questions concern axioms such as efficiency, strategyproofness, consistency, or fairness. In contrast, our agents' acceptable sets over lotteries are hidden halfspace cuts of the simplex, and the protocol never observes full rankings, approval sets, utilities, or thresholds. The problem is therefore not to design an axiomatic lottery rule from reported preferences, but to discover enough information, through binary accept/reject queries, to find a lottery accepted by every agent or determine that none exists. Geometrically, the consensus problem is a hidden halfspace-feasibility problem in the simplex. This perspective is related to Helly-type structure and Clarkson-style low-dimensional linear programming, LP-type problems, and violator spaces (Helly, 1923; Clarkson, 1995; Gärtner & Welzl, 2001; Gärtner et al., 2008). These methods also allow constraints to be accessed through queries. In our model, testing a specified agent at a candidate costs one query, and learning a sampled agent's inequality requires further queries. We account for both costs under limits on rounds and on the total number of queries per round.

Bei et al. (2015) study linear feasibility with unknown constraints, where a query returns the index of a constraint that the proposed point does not satisfy. Here the algorithm chooses which agent to query and receives a binary answer. Checking every agent at one lottery can therefore cost n queries, and must respect the batch cap.

Preference, utility, and reward elicitation in machine learning. Making a collective decision without fully learning every agent's preferences is a central aim of preference elicitation (Blum et al., 2004). A large body of work in the ML literature studies how to elicit or infer preferences and utilities while minimizing user burden. Constraint-based and regret-based utility elicitation ask targeted queries in order to recommend high utility decisions under partially known utilities (Boutilier et al., 2006; Braziunas & Boutilier, 2008; Viappiani & Boutilier, 2010). Inverse reinforcement learning and apprenticeship learning infer reward functions or policies from demonstrations, often using linear rewards or feature expectations (Ng & Russell, 2000; Abbeel & Ng, 2004; Ziebart et al., 2008). Preference-based reinforcement learning and RL from human feedback replace demonstrations or hand-coded rewards with comparisons or preference labels (Wirth et al., 2017; Sadigh et al., 2017; Christiano et al., 2017). Recent batch active preference-learning methods explicitly generate batches of preference queries to improve parallelization and human-query efficiency (Biyik et al., 2024). In our model, a stakeholder's accept/reject response is determined by an unknown linear score and threshold. The difference is that our goal is not to learn a reward model, rank alternatives, optimize a policy, or recommend a single user's preferred item. We study exact unanimous feasibility for multiple stakeholders, under worst-case hidden halfspace constraints, and we ask for precise query, round, and batch-size guarantees rather than Bayesian value-of-information, regret, or generalization guarantees.

Exact learning of geometric sets. Bshouty et al. (1998) study exact learning on finite grids using membership queries and equivalence queries, which can return a point where a proposed description is incorrect. Goldberg & Kwek (2000) study exact learning of rational polytopes from membership queries, given an interior point, and analyze the precision of the queried points. Abasi et al. (2016) give a two-round algorithm for learning a halfspace on $\{0,1\}^d$ with integer weights in $\{0,\dots,t\}$, using $d^{O(t)}$ membership queries. Our single-agent learning algorithm determines acceptance throughout the probability simplex. Our multi-agent algorithms find consensus or determine infeasibility under limits on rounds and a batch cap.

Active and batch active learning. Classical active learning studies how adaptively selected labels can reduce sample complexity relative to passive supervision (Cohn et al., 1994; Dasgupta, 2005; Balcan et al., 2006; Settles, 2009; Hanneke, 2014). Batch active learning selects several labels at once in order to amortize retraining, exploit parallel labeling, or avoid redundant acquisitions (Hoi et al., 2006; Guo & Schuurmans, 2007; Houlsby et al., 2011; Kirsch et al., 2019); related work on batch-mode active learning and information-parallel stochastic optimization gives approximation guarantees under adaptive-submodularity assumptions (Chen & Krause, 2013). The mathematical question in our work is different. Standard batch active learning usually optimizes acquisition quality

for classification or preference prediction, often under a data distribution or a posterior model. Our B-batched algorithms and lower bounds are worst-case, exact, and feasibility-oriented. They also distinguish two resources that are often conflated in ML practice: the number of adaptive rounds and the number of queries that can be issued in a round. In our multi-agent setting, a batch is not only a set of informative labels for one classifier; it is a scarce resource that must be split between learning geometric constraints and checking which agents can invalidate the current proposed lottery.

Limited interaction in fair division. Brânzei & Nisan (2019) study communication rounds and the number of bits exchanged in approximate cake cutting. Brânzei et al. (2020) study searching, sorting, and proportional cake cutting with a fixed number of query rounds. Their ordered-search analysis is closely related to our threshold searches. Our additional question is how to find a value accepted by all agents without learning every threshold, and how to divide a global query budget between agents and candidate values.

Bounded adaptivity, parallel queries, and geometric query complexity. Theoretical computer science has developed several models in which adaptivity, not just total query count, is the scarce resource. Examples include adaptivity hierarchies in property testing (Canonne & Gur, 2018), adaptive submodular optimization (Golovin & Krause, 2011), low-adaptivity submodular maximization (Balkanski & Singer, 2018; Balkanski et al., 2022), bounded adaptivity causal discovery (Choo & Shiragur, 2023), and round-limited query complexity for local search and Brouwer fixed points (Branzei & Li, 2022). These works share with ours the theme that reducing adaptive rounds can force more total queries. Our setting differs in the oracle and in the geometry: a query is an accept/reject evaluation of a lottery by a stakeholder, and the hidden object is an intersection of halfspaces over a simplex. We also separate two resource constraints that are distinct in elicitation systems: an R-round model with unlimited parallelism within a round and a B-batched model with a global per-round query cap.

B Additional Information for Section 2

B.1 Remarks on Expected Utility Assumption

This subsection clarifies the role of the expected utility representation $A_i = \{\mathbf{x} \in \Delta(S) : \langle u_i, \mathbf{x} \rangle \geq \tau_i\}$. The representation should be interpreted primarily as a structural assumption on the geometry of binary acceptability: each agent's acceptable set is a hidden halfspace cut of the simplex. It is not a full preference elicitation model. The protocol never asks an agent to report u_i, τ_i, a complete ranking, a cardinal utility scale, or a marginal rate of substitution. The only feedback is a membership query: given a lottery $\mathbf{x}$, is $\mathbf{x}$ acceptable?

Connection to query and active learning. Under this interpretation, our oracle model is a natural instance of membership query learning. Each queried lottery is selected by the algorithm, and the agent returns an accept/reject response. The unknown concept is a linear threshold function restricted to the simplex. Halfspaces and linear threshold functions are a canonical hypothesis class in learning theory, and membership queries are a canonical information model for concept learning (Angluin, 1988). The adaptive choice of queried points connects the model to active learning (Settles, 2009; Dasgupta, 2005); the round and batch-size constraints in our paper correspond to the practical limitations that human-in-the-loop systems often face when many queries must be prepared in advance or answered in parallel. The halfspace specialization is also closely related to generalized binary search and active learning of linear separators (Nowak, 2011).

This ML viewpoint is useful for interpreting the strength of the assumption. Our work does not claim that stakeholders consciously compute expected utilities. Rather, we assume that each agent's accept/reject responses are consistent with an unknown linear threshold rule on $\Delta(S)$. As in supervised and active learning, the class of possible rules is chosen to make sample or query complexity analysis possible; it need not describe how an agent reaches a decision.

Connection to reward and preference learning. The same linear form is standard in ML models of reward learning and preference learning (Boutilier et al., 2006; Viappiani & Boutilier, 2010; Biyik et al., 2024). Suppose each pure alternative s is represented by a feature vector $\phi(s)$, and agent i's latent score for a pure alternative is $\theta_i^\top \phi(s)$. A lottery $\mathbf{x}$ then induces the feature expectation $\mu(\mathbf{x}) = \sum_{j=1}^m x_j \phi(s_j)$ and the agent's score for the lottery is $\theta_i^\top \mu(\mathbf{x}) = \sum_{j=1}^m x_j \, \theta_i^\top \phi(s_j)$. Thus,

threshold acceptability takes the form $\theta_i^\top \mu(\mathbf{x}) \geq \tau_i$. In the tabular case where $\phi(s)$ is the basis vector for alternative s, this is exactly $\langle u_i, \mathbf{x} \rangle \geq \tau_i$.

This is the same modeling pattern used in inverse reinforcement learning and apprenticeship learning, where rewards are often represented as linear functions of features and policies are compared through their expected feature counts (Ng & Russell, 2000; Abbeel & Ng, 2004; Ziebart et al., 2008). It is also consistent with recent preference-based reward learning models, where a human's latent reward is learned from binary comparisons or preference labels (Sadigh et al., 2017; Christiano et al., 2017). Our setting differs in that the query is an accept/reject query for a lottery rather than a comparison between two trajectories, but the modeling abstraction is the same: a low-dimensional latent linear score induces binary feedback.

Operational interpretation of lotteries. The linearity assumption is particularly natural when a lottery is an operational randomization: for example, a randomized deployment policy, a traffic split between model variants, a mixture over interventions, a randomized assignment rule, or a distribution over fallback modes. If $u_i(s)$ represents a per-deployment benefit, safety margin, negative cost, compliance score, or risk-adjusted value of alternative s, then $\langle u_i, \mathbf{x} \rangle$ is the average score induced by implementing mixture $\mathbf{x}$. The threshold τ_i then represents a minimum acceptable standard.

This interpretation does not require a strong psychological claim that agents are risk-neutral over all consequences. Risk attitudes, safety margins, and application-specific transformations can be encoded in the option-level scores $u_i(s)$ or in the feature representation $\phi(s)$. What is assumed is that, after such encoding, the stakeholder's pass/fail rule is linear in the implemented mixture. Models with probability distortions, distributional constraints, or nonlinear ambiguity aversion are outside the present scope; they would require different concept classes and different query complexity analyses.

Normalization is not substantive. The bounds $u_i(s_j) \in [0, 1]$ and $\tau_i \in (0, 1]$ are only a normalization: every nontrivial halfspace cut of $\Delta(S)$ can be represented in this form. On the simplex, $\langle \mathbf{1}, \mathbf{x} \rangle = 1$ for every $\mathbf{x} \in \Delta(S)$. Therefore, adding the same constant to every coordinate of a linear score and multiplying by a positive scalar does not change the induced accept/reject set.

More explicitly, consider any nontrivial halfspace cut of the simplex, $H = \{\mathbf{x} \in \Delta(S) : \langle a, \mathbf{x} \rangle \geq b\}$, where "nontrivial" means that at least one point of the simplex is accepted and at least one point is rejected. Since a linear function on the simplex attains its minimum and maximum at vertices, nontriviality implies $\min_j a_j < b \leq \max_j a_j$. Choose $c > -\min_j a_j$. Then $\langle a, \mathbf{x} \rangle \geq b \iff \langle a + c\mathbf{1}, \mathbf{x} \rangle \geq b + c$ because $\langle \mathbf{1}, \mathbf{x} \rangle = 1$. Moreover, $a_j + c > 0$ for every j, and $b + c > 0$. Finally choose $\sigma > 0$ small enough that $\max_j \sigma(a_j + c) \leq 1$. Setting $u_j = \sigma(a_j + c)$ and $\tau = \sigma(b + c)$ gives us $u \in [0, 1]^m$, $\tau \in (0, 1]$, and exactly the same acceptable set. Thus, the mathematical content of the model is the hidden linear boundary, not the particular utility scale.

Already for one agent, no deterministic algorithm using finitely many queries on every instance can solve exact elicitation if acceptable sets may be arbitrary nonempty closed convex subsets of $\Delta(S)$. This class includes every singleton $A = \{\mathbf{x}\}$ with $\mathbf{x} \in \Delta(S)$. Such an algorithm has only countably many finite sequences of binary responses, and each sequence determines at most one output. It can therefore return only countably many distinct lotteries after finitely many queries. Correctness on every singleton would require returning every point of $\Delta(S)$, which is uncountable for $m \geq 2$. Thus finite-query exact elicitation requires a restriction on the acceptable sets or the domain, a different objective such as approximation, or a stronger oracle.

The halfspace model is therefore a deliberately clean benchmark: it is one of the simplest continuous convex concept classes for which exact membership query guarantees are meaningful. Our positive results use this structure, especially the fact that along an edge of the simplex membership changes at a single threshold and that learned constraints can be intersected offline as linear inequalities. The model should not be read as claiming universality for all stakeholder preferences.

Implications for the lower bounds. The halfspace assumption makes the lower bounds stronger, not weaker. The lower bounds are proved inside this restricted class. Consequently, any algorithm that is required to work for a richer acceptability model containing these halfspace instances is subject to the same lower bounds. In other words, even when each stakeholder's acceptability region is a single linear threshold, bounded adaptivity and bounded parallelism already create the costs identified in the paper.

Extensions. The model can already express some multi-criteria standards. If a stakeholder or review process has several separately checkable linear requirements, such as an expected cost upper bound, an expected risk upper bound, and an expected benefit lower bound, each requirement can be represented as a separate halfspace constraint. In the unanimity formulation, this is equivalent to adding one agent or reviewer constraint per requirement. If instead a stakeholder gives only aggregate accept/reject feedback for an unknown intersection of several hidden constraints, then the oracle model becomes one of hidden polyhedral acceptability. That is a natural extension, but it is a different learning problem from the hidden halfspace setting analyzed in our work.

B.2 Why Finite Precision Is Necessary

We recall the deterministic impossibility argument of Choo et al. (2026, Proposition B.1).

Consider $n = m = 2$. For each $\alpha \in (0, 1)$, let agent 1 have $u_1 = (0, 1)$ and $\tau_1 = \alpha$, and let agent 2 have $u_2 = (1, 0)$ and $\tau_2 = 1 - \alpha$. By the definition of $\mathbf{x}_{1,2,\beta}$ in Section 2, agent 1 accepts $\mathbf{x}_{1,2,\beta}$ if and only if $\beta \geq \alpha$, while agent 2 accepts it if and only if $\beta \leq \alpha$. Hence their unique consensus lottery is $\mathbf{x}_{1,2,\alpha}$.

Now fix a deterministic algorithm that makes finitely many queries on every instance. The algorithm has only countably many finite sequences of accept/reject responses, and its output is fixed by such a sequence. The instances above require a different output for every $\alpha \in (0, 1)$, which is an uncountable set. Therefore no deterministic finite-query algorithm can be correct on all of these instances. This proves the claimed impossibility when the parameters are unrestricted.

B.3 Running Example

For the instance in Section 2, agent 1 has $\alpha^1_{2,1} = \alpha^1_{3,1} = 0.2$, as illustrated in Figure 1(a). Agent 2 has edge threshold 0.3 on both edges directed toward $\mathbf{e}_2$, and agent 3 has edge threshold 0.5 on both edges directed toward $\mathbf{e}_3$. Agents 4 and 5 have edge threshold 0.1 on both edges directed toward $\mathbf{e}_1$. All these thresholds belong to Γ_ε.

To see why an unqueried agent still matters after $\mathbf{x}^*$ has been determined, change only agent 4's acceptance threshold τ_4 from 0.1 to 1. It then accepts only $\mathbf{e}_1$, so no consensus lottery exists. Both instances give identical answers to every query addressed to another agent. An exact deterministic algorithm therefore cannot distinguish them without querying agent 4. The same argument applies to agent 5. This is the checking cost captured by the term $(n - m)/B$ in Theorem 4.3.

C Proofs for Section 3

We first state two facts used by both fixed-round algorithms. The first uses the standard method of dividing an ordered candidate set into nearly equal parts with simultaneous queries (Brânzei et al., 2020); the same method is used for paths by Choo & Shiragur (2023). Here the number of parts may differ across rounds.

Lemma C.1. *Let Λ be a finite ordered set of size M, let $\lambda^* \in \Lambda$, and suppose a query at $\lambda \in \Lambda$ returns $\mathbf{1}[\lambda \geq \lambda^*]$. Let $a_1, \ldots, a_T$ be positive integers with $\prod_{r=1}^{T} a_r \geq M$. Then λ^* can be found in T rounds using at most $a_r - 1$ queries in round r.*

Proof. Start with $C_1 = \Lambda$. In round r, divide C_r into $\min\{a_r, |C_r|\}$ consecutive nonempty parts whose sizes differ by at most one. Query the largest element of every part except the last. The first accepted query identifies the part containing λ^*; if every query is rejected, λ^* lies in the last part. Keep that part as C_{r+1}. This uses at most $a_r - 1$ queries and gives $|C_{r+1}| \leq \lceil \frac{|C_r|}{a_r} \rceil$. For positive integers a, b, successive divisions with rounding up satisfy $\lceil \lceil M/a \rceil / b \rceil = \lceil M/(ab) \rceil$. Applying this identity over the T rounds implies $|C_{T+1}| \leq \lceil \frac{M}{\prod_{r=1}^{T} a_r} \rceil = 1$. The remaining candidate is therefore λ^*. □

The following reconstruction is from Choo et al. (2026, Algorithm 1 and Lemma 3.2).

Lemma C.2. *Fix an agent and omit its index. Let L_{acc} and L_{rej} be the indices of the accepted and rejected pure lotteries, and fix $\rho \in L_{\mathrm{rej}}$.*

If $\alpha_{\rho,j} = 1$ for every $j \in L_{\mathrm{acc}}$, then the vector $\mathbf{c}$ with $c_j = 1$ for $j \in L_{\mathrm{acc}}$ and $c_k = 0$ for $k \in L_{\mathrm{rej}}$ satisfies $A = \{\mathbf{x} \in \Delta(S) : \langle \mathbf{c}, \mathbf{x} \rangle \geq 1\}$.

Otherwise choose $a \in L_{\mathrm{acc}}$ with $\alpha_{\rho,a} < 1$. If the values $\alpha_{\rho,j}$ are known for all $j \in L_{\mathrm{acc}}$ and $\alpha_{k,a}$ are known for all $k \in L_{\mathrm{rej}} \setminus \{\rho\}$, then the vector $\mathbf{c}$ with

$$c_\rho = 0, \quad c_j = \frac{1}{\alpha_{\rho,j}} \text{ for all } j \in L_{\mathrm{acc}}, \quad c_k = \frac{1 - \alpha_{k,a} c_a}{1 - \alpha_{k,a}} \text{ for all } k \in L_{\mathrm{rej}} \setminus \{\rho\}$$

satisfies the same identity.

Proof. First suppose $\alpha_{\rho,j} = 1$ for every $j \in L_{\mathrm{acc}}$. Since $u(s_\rho) < \tau$, the threshold identity implies $u(s_j) = \tau$ for every accepted pure lottery. Every rejected pure lottery has utility strictly below τ. A lottery is therefore accepted exactly when it assigns probability zero to all rejected alternatives, or equivalently when $\sum_{j \in L_{\mathrm{acc}}} x_j = 1$. For the first vector in the statement, this is precisely $\langle \mathbf{c}, \mathbf{x} \rangle \geq 1$, since the inner product cannot exceed one.

Now suppose $\alpha_{\rho,a} < 1$. We show that the displayed coefficients are the original utilities, shifted by $u(s_\rho)$ and divided by the positive number $\tau - u(s_\rho)$. For every $j \in L_{\mathrm{acc}}$, the edge-threshold formula in the preliminaries gives

$$c_j = \frac{1}{\alpha_{\rho,j}} = \frac{u(s_j) - u(s_\rho)}{\tau - u(s_\rho)}.$$

Moreover, $\alpha_{\rho,a} < 1$ implies $u(s_a) > \tau$. Thus $0 < \alpha_{k,a} < 1$ for every $k \in L_{\mathrm{rej}}$, so each denominator $1 - \alpha_{k,a}$ in the statement is positive. The identity $\tau = (1 - \alpha_{k,a}) u(s_k) + \alpha_{k,a} u(s_a)$ now gives

$$c_k = \frac{1 - \alpha_{k,a} c_a}{1 - \alpha_{k,a}} = \frac{u(s_k) - u(s_\rho)}{\tau - u(s_\rho)} \quad \text{for all } k \in L_{\mathrm{rej}} \setminus \{\rho\}.$$

The same expression equals zero at $k = \rho$, as required. Since every $\mathbf{x} \in \Delta(S)$ satisfies $\sum_j x_j = 1$, we conclude that

$$\langle \mathbf{c}, \mathbf{x} \rangle = \frac{\langle u, \mathbf{x} \rangle - u(s_\rho)}{\tau - u(s_\rho)} \geq 1 \quad \Longleftrightarrow \quad \langle u, \mathbf{x} \rangle \geq \tau.$$

□

C.1 Proof of Theorem 3.1

Proof. We query every possible edge in the first round, but continue searching only on the edges needed for reconstruction. Let $H = |\Gamma_\varepsilon| = \mathcal{O}(\varepsilon^{-2})$.

For $R = 1$, query all pure lotteries and every $\mathbf{x}_{k,j,\gamma}$ with $k \neq j$ and $\gamma \in \Gamma_\varepsilon^-$. The pure-lottery answers identify L_{acc} and L_{rej}. On an edge from a rejected pure lottery to an accepted one, the smallest accepted queried value is the exact threshold. If all queried values below one are rejected, the threshold is one. Thus every threshold needed by Lemma C.2 is known. The lemma returns the exact acceptable set using $\mathcal{O}(m^2 H) = \mathcal{O}(m^2 \varepsilon^{-2})$ queries in this one batch.

For $R \geq 2$, set $c = \lceil (mH)^{1/R} \rceil$ and $t = \lceil \frac{H}{c^{R-1}} \rceil$. Then $1 \leq t \leq H$ and $t c^{R-1} \geq H$. In the first round, query all pure lotteries. For every ordered edge (k, j), partition Γ_ε into t consecutive parts of nearly equal size and query the largest element of every part except the last. Also query $\mathbf{x}_{k,j,1-\varepsilon}$. Repeated queries may be omitted.

After this round, choose $\rho \in L_{\mathrm{rej}}$. Every element of Γ_ε below one is at most $1 - \varepsilon$: if $p/q < 1$ and $q \leq 1/\varepsilon$, then $p/q \leq 1 - 1/q \leq 1 - \varepsilon$. Consequently, for each $j \in L_{\mathrm{acc}}$, the answer at $\mathbf{x}_{\rho,j,1-\varepsilon}$ tells us whether $\alpha_{\rho,j} < 1$ or $\alpha_{\rho,j} = 1$. If all these thresholds equal one, Lemma C.2 finishes without further queries.

Otherwise choose $a \in L_{\mathrm{acc}}$ with $\alpha_{\rho,a} < 1$. It remains to learn the thresholds on the edges (ρ, j) for which $\alpha_{\rho,j} < 1$, and on (k, a) for $k \in L_{\mathrm{rej}} \setminus \{\rho\}$. There are at most $m - 1$ such edges; every other threshold required by the reconstruction lemma is already known to be one. For each selected edge, the first-round partition answers leave at most $\lceil H/t \rceil \leq c^{R-1}$ consecutive candidates. Apply

Lemma C.1 to each of these candidate sets over rounds $2, \dots, R$, with split factor c in every round. These searches run in parallel and identify every remaining threshold. Lemma C.2 then returns the exact inequality.

The first round uses at most $m + m(m-1)t$ queries, and each later round uses at most $(m-1)(c-1)$ queries. The choices of c and t give

$$c - 1 < (mH)^{1/R}, \quad t \le H^{1/R} m^{-(R-1)/R} + 1.$$

Thus the largest batch has size $\mathcal{O}(m^2 + m^{1+1/R} H^{1/R})$, and the total query count is

$$\mathcal{O}(m^2 + R m^{1+1/R} H^{1/R}) = \mathcal{O}(m^2 + R m^{1+1/R} \varepsilon^{-2/R}).$$

□

C.2 Proof of Theorem 3.2

Proof. We use the first two rounds to choose the edges, leaving the remaining $R-2$ rounds for exact threshold searches. In round one, query all pure lotteries and obtain L_{acc} and L_{rej}. Choose $\rho \in L_{\mathrm{rej}}$. In round two, query $\mathbf{x}_{\rho,j,1-\varepsilon}$ for every $j \in L_{\mathrm{acc}}$. As in the preceding proof, this query is accepted exactly when $\alpha_{\rho,j} < 1$, or equivalently $u_i(s_j) > \tau_i$.

If no second-round query is accepted, every $\alpha_{\rho,j}$ equals one, and Lemma C.2 returns the inequality. Otherwise choose $a \in L_{\mathrm{acc}}$ whose second-round query was accepted. Search the edges (ρ, j) whose second-round query was accepted, and the edges (k, a) with $k \in L_{\mathrm{rej}} \setminus \{\rho\}$. There are at most $m-1$ such edges, and all their thresholds belong to Γ_ε^-.

Apply Lemma C.1 to these edges in parallel, using the common split factor $\lceil |\Gamma_\varepsilon^-|^{1/(R-2)} \rceil$ in each of rounds $3, \dots, R$. This identifies every remaining threshold, so the reconstruction lemma applies. Each of the first two rounds uses at most m queries, and each later round uses at most $(m-1)(\lceil |\Gamma_\varepsilon^-|^{1/(R-2)} \rceil - 1)$ queries. Since $|\Gamma_\varepsilon^-| = \mathcal{O}(\varepsilon^{-2})$, the batch size is $\mathcal{O}(m\varepsilon^{-2/(R-2)})$ and the total query count is $\mathcal{O}(mR\varepsilon^{-2/(R-2)})$. □

C.3 Proof of Theorem 3.3

Proof. We prove the dimension term and the numerical term on two families of valid agents. Their maximum is at least half their sum. Set $d = m - 1$ and $Q = 1/\varepsilon$.

The dimension term. For each $a = (a_1, \dots, a_d) \in [Q]^d$, take

$$u^{(a)}(s_1) = 0, \quad u^{(a)}(s_{j+1}) = a_j \varepsilon \quad \text{for all } j \in [d], \quad \tau^{(a)} = \varepsilon.$$

These parameters are quantized, and the agent rejects $\mathbf{e}_1$ and accepts every other pure lottery. Its acceptable set is

$$A^{(a)} = \left\{ \mathbf{x} \in \Delta(S) : \sum_{j=1}^{d} a_j x_{j+1} \ge 1 \right\}.$$

The Q^d sets are pairwise distinct: on the edge from $\mathbf{e}_1$ to $\mathbf{e}_{j+1}$, the threshold is $1/a_j$.

A batch of q arbitrary lottery queries gives at most $D_{q,d} = \sum_{k=0}^{d} \binom{q}{k}$ different answer vectors on this family, with $\binom{q}{k} = 0$ for $k > q$. To see this, discard repeated queries and the constantly rejected query at $\mathbf{e}_1$. Each remaining query determines an affine hyperplane $\sum_j a_j x_{j+1} = 1$ in the space of coefficient vectors a. If a coefficient vector lies on some of these hyperplanes, increasing all its coordinates by a sufficiently small common positive amount makes every equality strict on the accepted side, while preserving all other answers. This is because the query coordinates are nonnegative and not all of $x_2, \dots, x_m$ are zero. Thus every realized answer vector also occurs in an open region of the hyperplane arrangement, even though the shifted vector need not belong to the grid. The region count is at most $D_{q,d}$ (Buck, 1943).

We next translate this bound on answers into a bound on expected queries. Fix a deterministic R-round tree that is correct on $K \ge 1$ members of the family. They reach distinct leaves, because

their acceptable sets differ. For a correct leaf λ, let $q_r(\lambda)$ be the number of queries in round r, padding a shorter execution with empty rounds, and set $P_\lambda = \prod_{r=1}^{R} D_{q_r(\lambda),d}$ and $q(\lambda) = \sum_{r=1}^{R} q_r(\lambda)$. Give the root weight one. At a node issuing q queries, give each realized child the parent's weight divided by $D_{q,d}$. There are at most $D_{q,d}$ children, so the weights of the correct leaves sum to at most one: $\sum_\lambda P_\lambda^{-1} \le 1$. The arithmetic–geometric mean inequality therefore implies $K \le (\prod_\lambda P_\lambda)^{1/K}$.

For all $q \ge 0$, $D_{q,d} \le [e(1 + \frac{q}{d})]^d$. For $q \le d$, use $D_{q,d} = 2^q \le e^d$; for $q > d$, use $\sum_{k=0}^{d} \binom{q}{k} \le (eq/d)^d$. Taking logarithms and using their concavity, first over rounds and then over the K correct leaves, gives

$$\log K \le \frac{1}{K} \sum_\lambda \log P_\lambda \le dR \log \left[e \left(1 + \frac{\sum_\lambda q(\lambda)}{KdR} \right) \right].$$

Hence the average number of queries over the correct inputs is at least

$$\frac{1}{K} \sum_\lambda q(\lambda) \ge dR \left(\frac{K^{1/(dR)}}{e} - 1 \right).$$

For a randomized algorithm, fix its random choices and let K be the number of inputs it answers correctly. Under the uniform distribution on $[Q]^d$, the average query count for these fixed choices is at least

$$\frac{dR}{Q^d} \left(\frac{K^{1+1/(dR)}}{e} - K \right),$$

where the expression is zero when $K = 0$. Success probability at least $2/3$ on every input gives $\mathbb{E}[K] \ge 2Q^d/3$, while $K \le Q^d$. By convexity of the power function, averaging the last display over the random choices yields at least

$$dR \left(\frac{(2/3)^{1+1/(dR)}}{e} Q^{1/R} - 1 \right) = \Omega(dRQ^{1/R})$$

queries when $Q^{1/R}$ is sufficiently large. Thus some input has at least this expected cost. Deterministic exact algorithms are included as the case with no internal randomness.

The numerical term. For each $p/q \in \Gamma_\varepsilon$ in lowest terms, take $u(s_2) = q\varepsilon$, $u(s_j) = 0$ for $j \ne 2$, and $\tau = p\varepsilon$. This valid nontrivial agent accepts an arbitrary lottery exactly when $x_2 \ge p/q$. There are $|\Gamma_\varepsilon| = \Theta(Q^2)$ distinct acceptable sets, and q_r queries in a round give at most $q_r + 1$ answer vectors. Use the preceding leaf-weight calculation with factors $q_r + 1$ in place of $D_{q_r,d}$. It gives average query count at least $R(K^{1/R} - 1)$ over any $K \ge 1$ correctly answered inputs. Averaging over all inputs and the random choices exactly as above gives $\Omega(R|\Gamma_\varepsilon|^{1/R}) = \Omega(RQ^{2/R})$ queries for fixed R and sufficiently large Q. Combining the two lower bounds, substituting $Q = 1/\varepsilon$, and using $d \ge m/2$ proves the theorem. □

C.4 Proof of Theorem 3.4

Proof. Before global round r, each copy of `LearnHalfspace`(i) has received exactly its own answers from the preceding rounds. Its next batch therefore depends only on answers from earlier global rounds, not on answers from round r. Taking the union of these batches is a valid round. Inductively, every copy proceeds exactly as it would when run alone and returns an exact inequality for its agent. Consequently, $\bigcap_{i \in N} A_i = \{\mathbf{x} \in \Delta(S) : \langle \mathbf{c}_i, \mathbf{x} \rangle \ge 1 \text{ for every } i \in N\}$. By its definition, `Select` returns a lottery in this intersection when it is nonempty and returns NULL otherwise.

Every copy finishes within R rounds, contributes at most B' queries to any one batch, and uses at most Q' queries in total. Summing over the n copies gives at most R rounds, batch size nB', and nQ' total queries. □

C.5 An instance-sensitive rejector method

We modify Algorithm 2 of Choo et al. (2026) by learning all agents that reject the current lottery in parallel. The proof below uses Helly's theorem to find a small set of agents that determines the full answer, and shows that each rejected candidate leads to learning a new agent from this set (Helly, 1923; Clarkson, 1995).

Proposition C.3. *Fix $R \geq 1$. Suppose each call to* `LearnHalfspace`(i) *is an exact R-round algorithm using at most Q' queries in total and at most B' queries in one round. Then* `LearnFromRejectors` *returns a consensus lottery when one exists and returns* NULL *otherwise. It uses at most $m(R+1)$ query rounds, batch size at most $\max\{n, nB'\}$, and at most $nm + nQ'$ queries.*

For a feasible instance, let κ be the smallest number of agents whose intersection has the same lexicographically maximum point as the full intersection. For an infeasible instance, let κ be the smallest number of agents whose intersection is empty. In the two cases, $\kappa \leq m-1$ and $\kappa \leq m$, respectively. Let L be the number of halfspaces actually learned.

Corollary C.4. *Under the assumptions of Proposition C.3, on a feasible instance* `LearnFromRejectors` *uses at most $\kappa(R+1)+1$ rounds and $n(\kappa+1)+LQ'$ queries. On an infeasible instance it uses at most $\kappa(R+1)$ rounds and $n\kappa + LQ'$ queries.*

Algorithm 2 `LearnFromRejectors`(R)

Input: $N = [n]$, $S = \{s_1, \ldots, s_m\}$, ε, integer $R \geq 1$
$C \leftarrow \varnothing$ and $N' \leftarrow N$
while True **do**
  $\mathbf{x} \leftarrow$ `Select`(C)
  if $\mathbf{x} =$ NULL **then return** NULL
  end if
  Query `Query`$(i, \mathbf{x})$ for all $i \in N'$ in one parallel round
  $V \leftarrow \{i \in N' : \texttt{Query}(i, \mathbf{x}) = \texttt{False}\}$ ▷ rejecting agents
  if $V = \varnothing$ **then return** $\mathbf{x}$
  end if
  (In parallel over $i \in V$**)** Run `LearnHalfspace`(i) and obtain $(\mathbf{c}_i)_{i \in V}$
  $C \leftarrow C \cup \{\mathbf{c}_i : i \in V\}$ and $N' \leftarrow N' \setminus V$
end while

Proof of Proposition C.3. Every learned agent accepts the candidate returned by `Select`(C), because its exact inequality belongs to C. If all remaining agents accept the candidate as well, the returned lottery is therefore a consensus lottery. If `Select`$(C) =$ NULL, even the learned constraints have empty intersection, so the full instance is infeasible. It remains to bound the number of checks before one of these outcomes occurs.

First suppose the full intersection is nonempty, and let $\mathbf{x}^*$ be its lexicographically maximum lottery. The lotteries in $\Delta(S)$ that are lexicographically larger than $\mathbf{x}^*$ form a convex set. Indeed, in a strict convex combination of two such lotteries, the first coordinate at which either lottery differs from $\mathbf{x}^*$ is larger than the corresponding coordinate of $\mathbf{x}^*$. This convex set has empty intersection with all the A_i together. Apply Helly's theorem (Helly, 1923) in the $(m-1)$-dimensional hyperplane $\sum_j x_j = 1$. An empty subintersection of at most m sets must include the set of lexicographically larger lotteries, since every A_i contains $\mathbf{x}^*$. Thus there is a set $W \subseteq N$ of at most $m-1$ agents whose intersection contains $\mathbf{x}^*$ and no lexicographically larger lottery. Equivalently,

$$\texttt{Select}(\{\mathbf{c}_i : i \in W\}) = \mathbf{x}^*, \quad |W| \leq m-1. \tag{1}$$

Here an intersection over no agents means $\Delta(S)$, as in the definition of `Select`$(\varnothing)$.

Consider a check at $\mathbf{x} =$ `Select`(C). The lottery $\mathbf{x}^*$ satisfies every learned constraint, so $\mathbf{x}$ is lexicographically no smaller than $\mathbf{x}^*$. If every agent in W accepted $\mathbf{x}$, then equation 1 would also make $\mathbf{x}$ lexicographically no larger than $\mathbf{x}^*$. We would have $\mathbf{x} = \mathbf{x}^*$, and every agent would accept. Hence whenever the rejecting set V is nonempty, it contains an agent of W. This agent is not yet learned, since every learned agent accepts $\mathbf{x}$. Learning all of V therefore learns a new member of W. There are at most $|W| \leq m-1$ such checks, followed by at most one check in which every remaining agent accepts. This also covers $W = \varnothing$, when the first candidate is already $\mathbf{x}^*$.

Now suppose the full intersection is empty. Helly's theorem gives $H \subseteq N$ with $|H| \leq m$ and $\bigcap_{i \in H} A_i = \varnothing$. Every candidate is rejected by some agent in H, and that agent cannot already have been learned. Each check therefore leads to learning a new member of H. After at most $|H|$ such checks and the corresponding learning calls, the learned constraints have empty intersection. The next application of `Select` returns NULL without another query.

In either case there are at most m checking rounds. Each uses at most n queries, and each nonempty rejecting set is learned in parallel in at most R further rounds, as in Theorem 3.4. The total is therefore at most $m(R+1)$ rounds. Checks use at most nm queries altogether; each agent is learned at most once, adding at most nQ' queries. A checking batch has size at most n, while a learning batch has size at most nB'. These are the claimed bounds. □

Proof of Corollary C.4. On a feasible instance, choose a smallest set W of κ agents whose intersection has the same lexicographically maximum point as the full intersection. The preceding proof shows that each check with a rejection leads to learning a previously unlearned member of W. There are therefore at most κ such checks, each followed by at most R learning rounds, and at most one final accepting check. This gives $\kappa(R+1)+1$ rounds. The checks use at most $n(\kappa+1)$ queries, and learning the L agents uses at most LQ' more.

On an infeasible instance, choose a smallest set H of κ agents with empty intersection. Each check leads to learning a new member of H. After at most κ checks and their learning calls, `Select` returns NULL without another query batch. The corresponding bounds are $\kappa(R+1)$ rounds and $n\kappa + LQ'$ queries. □

C.6 One-round randomized complexity

Theorem C.5. *Fix $m = 2$ and ε-quantized parameters. For every $n \geq 2$, the one-round query complexity of consensus elicitation is $\Theta(n\varepsilon^{-2})$. The lower bound holds for randomized algorithms that are correct with probability at least $2/3$ on every instance, measured in worst-case expected queries.*

Proof. For the lower bound, let $M = |\Gamma_\varepsilon^-|$ and split the agents into a lower-bound group L and an upper-bound group U, of sizes $p = \lfloor n/2 \rfloor$ and $q = n - p$. For every $(i, j, \alpha) \in L \times U \times \Gamma_\varepsilon^-$, let agent i require $\beta \geq \alpha$ and agent j require $\beta \leq \alpha$ in the lottery $\mathbf{x}_{1,2,\beta}$. All other lower-bound agents require $\beta \geq \varepsilon$, and all other upper-bound agents require $\beta \leq 1 - \varepsilon$. These are valid quantized constraints. In particular, for $\alpha = a/d$ in lowest terms, utilities $(0, d\varepsilon)$ and threshold $a\varepsilon$ give the lower bound; utilities $(d\varepsilon, 0)$ and threshold $(d-a)\varepsilon$ give the upper bound. Since $\varepsilon \leq \alpha \leq 1 - \varepsilon$, this instance has the unique consensus lottery $\mathbf{x}_{1,2,\alpha}$.

Fix a deterministic one-round batch containing T queries. Let a_i and b_j be the numbers of distinct query coordinates addressed to lower-bound agent i and upper-bound agent j, respectively. For a fixed pair (i, j), only these two agents' answers depend on α. Their joint answer vector can change only when α crosses or equals one of their queried coordinates, so it takes at most $2(a_i + b_j) + 1$ values. A single output cannot be correct for two different values of α. Thus, over all indexed triples (i, j, α), the batch is correct on at most

$$\sum_{i \in L} \sum_{j \in U} (2(a_i + b_j) + 1) \leq 2 \max\{p, q\} T + pq$$

triples, because $\sum_i a_i + \sum_j b_j \leq T$.

For a randomized one-round algorithm, the batch depends only on its internal randomness, not on the instance. Success probability at least $2/3$ on every indexed instance implies that the expected number of correctly answered triples is at least $2pqM/3$. Averaging the preceding bound over the random choices gives $\mathbb{E}[T] \geq \frac{pq(2M/3-1)}{2\max\{p,q\}}$. For $M \geq 2$, this is $\Omega(nM) = \Omega(n\varepsilon^{-2})$. If $M = 1$, then $\varepsilon = 1/2$; the agent-checking argument in Theorem C.6 gives the required $\Omega(n)$ bound.

For the upper bound, query every agent at both pure lotteries and at every $\mathbf{x}_{1,2,\gamma}$ with $\gamma \in \Gamma_\varepsilon^-$, all in one round. Nontriviality implies that exactly one pure lottery is accepted. The direction and the endpoint of each agent's acceptable interval are therefore determined, including an upper endpoint of zero or a lower endpoint of one. Intersect the intervals and return any lottery in the intersection, or NULL if it is empty. This uses $\mathcal{O}(n|\Gamma_\varepsilon|) = \mathcal{O}(n\varepsilon^{-2})$ queries. □

C.7 Randomized lower bounds for fixed rounds

Theorem C.6. *Under ε-quantized parameters, for every $n \geq 2$ and every fixed integer $R \geq 1$, every R-round consensus algorithm has worst-case query complexity $\Omega(n + R\varepsilon^{-2/R})$, even when $m = 2$.*

The same lower bound holds for the worst-case expected query complexity of randomized algorithms that are correct with probability at least $2/3$ on every instance.

Proof. We prove the numerical term and the agent-checking term separately.

The numerical term. Use two agents with opposite inequalities at the same unknown $\alpha \in \Gamma_\varepsilon^-$, as in the preceding proof. Agent 1 requires $\beta \geq \alpha$ and agent 2 requires $\beta \leq \alpha$. Add $n-2$ agents that require $\beta \geq \varepsilon$. The unique consensus lottery is $\mathbf{x}_{1,2,\alpha}$, and the added agents' answers do not depend on α. Let $M = |\Gamma_\varepsilon^-|$. This is an ordered-search problem, but we must count equality answers and allow the number of queries to depend on earlier answers (Brânzei et al., 2020).

A batch of q queries gives at most $2q+1$ answer vectors as α varies: answers can change only at a queried coordinate or on crossing it. The leaf-weight calculation in the proof of Theorem 3.3, now with factors $2q_r+1$, shows that a deterministic R-round tree correct on $K \geq 1$ of these inputs uses at least $\frac{R}{2}(K^{1/R}-1)$ queries on average over the K correct inputs. The factor $1/2$ follows from $\prod_{r=1}^{R}(2q_r+1) \leq (1+2\sum_r q_r/R)^R$. Consequently, its average query count under the uniform distribution on all M inputs is at least $\frac{R}{2M}(K^{1+1/R}-K)$ with value zero when $K=0$.

For a randomized algorithm, let K vary with its internal randomness. We have $\mathbb{E}[K] \geq 2M/3$ and $K \leq M$. Convexity of $K \mapsto K^{1+1/R}$ therefore gives expected query count at least

$$\frac{R}{2}\left((2/3)^{1+1/R}M^{1/R}-1\right) = \Omega(R\varepsilon^{-2/R})$$

for fixed R and sufficiently small ε. In particular, some input has at least this expected cost.

The agent-checking term. Fix $\alpha = \varepsilon$ in the same construction. For each additional agent $\ell \geq 3$, change only that agent so that it accepts exactly $\mathbf{e}_1$, using utilities $(1,0)$ and threshold one. The original instance has unique consensus $\mathbf{x}_{1,2,\varepsilon}$, whereas the changed instance is infeasible.

Run the algorithm on these two instances with the same internal randomness. Until it queries agent ℓ, the answers, later queries, and eventual output are identical. On the event that it never queries ℓ, it therefore cannot be correct on both instances. The two error probabilities sum to at most $2/3$, so the probability of querying ℓ on the original feasible instance is at least $1/3$. Summing over the additional agents gives at least $(n-2)/3$ expected queries, which is $\Omega(n)$ for $n \geq 3$. For $n=2$, compare the instance in which both agents accept only $\mathbf{e}_1$ with the instance in which both accept only $\mathbf{e}_2$. The same argument shows that a query is made with probability at least $1/3$, giving the required constant bound.

The worst-case expected cost is at least the larger of these two bounds, and hence at least a constant multiple of their sum. For the bounded remaining values of $1/\varepsilon$, the checking term also covers the numerical term after adjusting the constant for the fixed R. This proves $\Omega(n + R\varepsilon^{-2/R})$, including for deterministic exact algorithms. □

C.8 Two-alternative deterministic complexity and three-round randomization

We first give the deterministic lower bound used both here and in the batch-limited result. It keeps uncertainty about the value of consensus and about which agents determine it.

Lemma C.7. *Let $R \geq 1$ be an integer and $p = \lfloor n/2 \rfloor$. Every deterministic exact two-alternative consensus algorithm using at most R rounds and at most q queries on every instance satisfies $q \geq \frac{p}{2}((|\Gamma_\varepsilon^-|/2)^{1/R}-1)$. This holds even if every agent's inequality direction is given to the algorithm.*

Proof. We choose answers so that many agents can still impose any threshold in the same consecutive set of candidates. Two candidates remaining after the last round will then require different outputs on instances with identical answers.

Use p lower-bound agents and $n-p$ upper-bound agents, and give their directions to the algorithm. Every threshold $a/d \in \Gamma_\varepsilon^-$ is valid: use utilities $(0, d\varepsilon)$ and threshold $a\varepsilon$ for a lower bound, and utilities $(d\varepsilon, 0)$ and threshold $(d-a)\varepsilon$ for an upper bound. Suppose, for a contradiction, that q is smaller than the claimed bound. Initially let $C = \Gamma_\varepsilon^-$ and leave every agent's threshold unfixed.

Maintain the following two properties: every unfixed threshold may independently be any element of C, consistently with all earlier answers; and every fixed agent accepts every value in C.

In round r, let q_r be the number of queries in the chosen batch and set $K_r = \max\{1, \lceil \frac{2Rq_r}{p} \rceil\}$. Partition C into K_r consecutive nonempty parts of nearly equal size. We justify below that this partition is always possible. The closed intervals from the first to the last candidate in these parts are disjoint. Each query coordinate belongs to at most one of them. Some part C' therefore has at most $q_r/K_r \le p/(2R)$ queries to unfixed agents within its closed interval; if $q_r = 0$, there are none.

For every unfixed agent addressed by such a query, fix its threshold to $\min C$ for a lower-bound agent and to $\max C$ for an upper-bound agent. These choices agree with all earlier answers and accept every value in C'. At most $p/(2R)$ agents are fixed in this round. Every other unfixed agent has no current query between $\min C'$ and $\max C'$, including the endpoints. Its current answers are therefore identical for all thresholds in C'. Give these answers, answer fixed agents according to their fixed thresholds, and replace C by C'. Both maintained properties continue to hold.

To justify the partitions, the preceding answers can always be realized by setting all unfixed thresholds to any one value in the current C. The batch chosen after those answers must therefore obey $\sum_{s=1}^{r} q_s \le q$. Since $K_s \le 1 + 2Rq_s/p$, the arithmetic–geometric mean inequality, with $R - r$ additional factors equal to one, gives

$$\prod_{s=1}^{r} K_s \le \left(1 + \frac{2\sum_{s=1}^{r} q_s}{p}\right)^R \le \left(1 + \frac{2q}{p}\right)^R < \frac{|\Gamma_\varepsilon^-|}{2}.$$

Before round r, the candidate count is at least $\lfloor |\Gamma_\varepsilon^-| / \prod_{s<r} K_s \rfloor$, by successive division with rounding down. The last display makes this count at least $2K_r$, so the proposed partition is possible. It also leaves at least two candidates after round R. A shorter execution may be padded with empty rounds.

At most $p/2$ agents have been fixed over all rounds. Since each direction initially has at least p agents, there is still an unfixed agent of each direction. Choose distinct $\alpha, \alpha' \in C$. Setting all unfixed thresholds to α gives a valid instance with unique consensus $\mathbf{x}_{1,2,\alpha}$; setting them all to α' gives one with unique consensus $\mathbf{x}_{1,2,\alpha'}$. Fixed agents accept both values, and the maintained properties ensure that both instances give every answer already supplied. The algorithm has the same output on the two instances and cannot be correct on both. This contradiction proves the lemma. □

Proof of Theorem 3.5. For fixed R, Lemma C.7 and $|\Gamma_\varepsilon^-| = \Theta(\varepsilon^{-2})$ give the deterministic lower bound $\Omega_R(n\varepsilon^{-2/R})$ for sufficiently small ε. The checking bound in Theorem C.6 covers the remaining bounded values of $1/\varepsilon$, with a constant that may depend on R.

For the deterministic upper bound, every endpoint belongs to $\{0\} \cup \Gamma_\varepsilon$. For each agent, divide the remaining endpoint candidates into nearly equal consecutive parts, with split factor $\lceil (|\Gamma_\varepsilon| + 1)^{1/R} \rceil$ in each round. Place each query strictly between adjacent parts, rather than at a candidate endpoint. In the first round, also query $\mathbf{e}_1$ to determine the agent's direction. Once that answer is known, the other first-round answers identify the part containing its endpoint, regardless of direction. The same searches continue in later rounds with the directions known. The candidate-count calculation in Lemma C.1 shows that all endpoints are identified within R rounds, using at most $n + nR(\lceil (|\Gamma_\varepsilon| + 1)^{1/R} \rceil - 1) = \mathcal{O}_R(n\varepsilon^{-2/R})$ queries. Intersecting the learned intervals solves consensus.

For the randomized upper bound, let $k = \lceil \sqrt{n} \rceil$. In the first round, query every agent at $\mathbf{e}_1$ and choose a uniform sample of k agents without replacement. In that same round, query each sampled agent at every $\mathbf{x}_{1,2,\beta}$ with $\beta \in \{0\} \cup \Gamma_\varepsilon$. These answers determine all directions and the sampled endpoints exactly. Let $[\ell, u]$ be the intersection of the sampled acceptable intervals, using zero or one for a missing lower or upper bound. If it is empty, return NULL.

In the second round, query every lower-bound agent at $\mathbf{x}_{1,2,\ell}$ and every upper-bound agent at $\mathbf{x}_{1,2,u}$. An agent accepting this test accepts the whole interval $[\ell, u]$, so remove it. In the third round, learn every remaining endpoint by querying all candidate values as in round one. Intersect these remaining constraints with $[\ell, u]$ and return any lottery in the intersection, or NULL if it is empty. Every possible consensus value lies in $[\ell, u]$, and every removed agent accepts that interval. The final intersection is therefore exactly the full feasible interval; no additional checking round is needed.

Lemma D.6 bounds the expected number of agents remaining after the second round by $2(n-k)/(k+1)$. The expected number of queries is consequently

$$\mathcal{O}\left(n + (|\Gamma_\varepsilon| + 1)\left(k + \frac{n}{k+1}\right)\right) = \mathcal{O}(n + \sqrt{n}\,\varepsilon^{-2}).$$

Every choice of the sample gives a correct answer within three rounds. For $n = \varepsilon^{-4}$, the randomized upper bound and the checking lower bound give $\Theta(n)$ expected queries, whereas the deterministic three-round bound is $\Theta(n\varepsilon^{-2/3}) = \Theta(n^{7/6})$. □

D Proofs for Section 4

Throughout this section, $Q := 1/\varepsilon$, and $|\Gamma_\varepsilon| = \mathcal{O}(Q^2)$. We take the batch cap B to be a positive integer; replacing a real cap by $\lfloor B \rfloor$ preserves the allowed batches and all stated asymptotic bounds.

D.1 Threshold search on one edge

Proof of Lemma 4.1. For the upper bound, apply Lemma C.1 with split factor $B+1$ in every round. After $T = \lceil \log_{B+1} M \rceil$ rounds, only one candidate remains, and at most BT queries have been used. When $M = 1$, no queries or rounds are needed.

For the lower bound, sort the distinct query values in a batch. Their answers have the form $0, \dots, 0, 1, \dots, 1$, so at most B queries give at most $B+1$ different answer vectors. A deterministic tree of depth at most T therefore has at most $(B+1)^T$ leaves. Distinct thresholds require distinct outputs, hence distinct leaves. Thus $(B+1)^T \geq M$, or $T \geq \lceil \log_{B+1} M \rceil$. □

D.2 One-agent upper bound

Theorem D.1. *Fix $B \geq 1$ and integers $g, b \geq 1$ with $g \leq m-1$ and $gb \leq B$. One hidden halfspace can be learned exactly in at most $2\lceil \frac{m}{B} \rceil + \lceil \frac{m-1}{g} \rceil \lceil \log_{b+1} |\Gamma_\varepsilon^-| \rceil$ rounds and at most $2m + (m-1)b\lceil \log_{b+1} |\Gamma_\varepsilon^-| \rceil$ queries.*

Proof. First identify the edges required by Lemma C.2. Query all pure lotteries in $\lceil m/B \rceil$ rounds, obtaining L_{acc} and L_{rej}, and choose $\rho \in L_{\mathrm{rej}}$. For every $j \in L_{\mathrm{acc}}$, query $\mathbf{x}_{\rho,j,1-\varepsilon}$. These tests take at most another $\lceil m/B \rceil$ rounds. As shown in the fixed-round proofs, a test is accepted exactly when $\alpha_{\rho,j} < 1$.

If all tests are rejected, the reconstruction lemma already determines the acceptable set. Otherwise choose $a \in L_{\mathrm{acc}}$ whose test was accepted. The thresholds still needed are those on (ρ, j) with $\alpha_{\rho,j} < 1$, and on (k, a) with $k \in L_{\mathrm{rej}} \setminus \{\rho\}$. There are at most $m-1$ such thresholds, all in Γ_ε^-. Every other threshold used in the reconstruction is already known to be one.

Divide the remaining edge searches into groups of at most g edges. For each group, search its edges in parallel with at most b queries per edge per round. The batch size is at most $gb \leq B$, and Lemma 4.1 identifies every threshold in the group within $\lceil \log_{b+1} |\Gamma_\varepsilon^-| \rceil$ rounds. Once all groups have been searched, Lemma C.2 returns the exact inequality.

The initial steps use at most $2\lceil m/B \rceil$ rounds and $2m$ queries. There are at most $\lceil (m-1)/g \rceil$ groups, and each of the at most $m-1$ edges uses at most $b\lceil \log_{b+1} |\Gamma_\varepsilon^-| \rceil$ queries. Adding these costs gives both stated bounds. □

D.3 A tree lemma for randomized lower bounds

Lemma D.2. *Let a deterministic rooted tree have at most $D \geq 2$ children at every node. If it is correct on $K \geq 1$ inputs and no one output is correct for two of them, then the average depth of the corresponding K leaves is at least $\log_D K$.*

Consequently, suppose a randomized algorithm is correct with probability at least $2/3$ on each of M inputs for which no one output is correct on two inputs, and every fixed random seed gives such a tree. Under the uniform distribution on the M inputs, its expected depth is $\Omega(\log_D M)$.

Proof. The K correct inputs reach distinct leaves. If their depths are $d_1, \ldots, d_K$, then $\sum_{j=1}^{K} D^{-d_j} \leq 1$: starting with weight one at the root and dividing by D at each step assigns these weights to the leaves, whose total cannot exceed one. This is Kraft's inequality (Kraft, 1949). Convexity of $x \mapsto D^{-x}$ gives

$$D^{-\frac{1}{K}\sum_j d_j} \leq \frac{1}{K}\sum_j D^{-d_j} \leq \frac{1}{K},$$

so the average depth is at least $\log_D K$.

For the randomized claim, an infinite expected depth already satisfies the lower bound, so assume the expectation is finite. For almost every fixed choice of the randomness, all of the finitely many inputs terminate. Let K be the number of them answered correctly for these choices. The average depth over all M inputs is at least $(K/M)\log_D K$, with $0 \log_D 0 = 0$. Moreover, $\mathbb{E}[K] \geq 2M/3$. For $M \geq 2$, convexity of $K \mapsto K \log_D K$ and its monotonicity for $K \geq 1$ gives us

$$\mathbb{E}[\text{depth}] \geq \frac{1}{M}\mathbb{E}[K \log_D K] \geq \frac{2}{3}\log_D(2M/3) = \Omega(\log_D M).$$

For $M = 1$, the asserted lower bound is zero and is immediate. □

D.4 One-agent lower bound

Theorem D.3. *Let $d = m - 1$ and $D_{B,d} := \sum_{k=0}^{d} \binom{B}{k}$, $\binom{B}{k} := 0$ for $k > B$. There is a family of Q^d pairwise distinct one-agent halfspaces such that every B-batched algorithm with error at most $1/3$ has worst-case expected round complexity $\Omega(\frac{d \log Q}{\log D_{B,d}})$. The same bound holds for deterministic exact algorithms.*

Proof. Use the family $A^{(a)}$ from the proof of Theorem 3.3, with $a \in [Q]^d$: $A^{(a)} = \{\mathbf{x} \in \Delta(S) : \sum_{j=1}^{d} a_j x_{j+1} \geq 1\}$. That proof establishes both that these are Q^d distinct valid quantized acceptable sets and that any batch of q arbitrary lottery queries has at most $\sum_{k=0}^{d} \binom{q}{k}$ realized answer vectors. In particular, the count includes equality answers, since they can be reproduced by increasing all coefficients slightly. For $q \leq B$, the number of answers is at most $D_{B,d}$.

Fixing the algorithm's internal randomness therefore gives a tree with at most $D_{B,d}$ children per node on this family. No output inequality can represent two different acceptable sets. Apply Lemma D.2 to the Q^d inputs. The average expected depth is $\Omega(\log(Q^d)/\log D_{B,d})$, and the worst-case expected number of rounds is at least this average. This proves the bound for randomized algorithms with error at most $1/3$, and hence for deterministic exact algorithms as well. □

D.5 Proof of Theorem 4.2

Proof. **Upper bounds on rounds.** For $1 \leq B \leq m - 1$, apply Theorem D.1 with $g = B$ and $b = 1$. Since $|\Gamma_\varepsilon^-| = \mathcal{O}(\varepsilon^{-2})$, the bound is $\mathcal{O}((m/B)\log(1/\varepsilon))$ rounds. For $B \geq m - 1$, choose $g = m - 1$ and $b = \lfloor B/(m-1) \rfloor$. All remaining edges are then searched together, giving $\mathcal{O}(1 + \frac{\log(1/\varepsilon)}{\log(1+B/m)})$ rounds. Here $\lfloor B/(m-1) \rfloor \geq B/[2(m-1)]$, so the logarithmic denominators differ only by a constant factor.

Lower bounds on rounds. Let $d = m - 1$, so $d \leq m \leq 2d$. For $B \leq d$, the quantity in Theorem D.3 is $D_{B,d} = 2^B$. The theorem therefore gives $\Omega((m/B)\log(1/\varepsilon))$ expected rounds. For $B \geq d$, use $D_{B,d} \leq (eB/d)^d$ to obtain $\Omega(\frac{\log(1/\varepsilon)}{\log(1+B/m)})$. There is also an $\Omega(1)$ lower bound. Choose two distinct halfspaces from the family. With the same internal randomness, an execution making no queries has the same output on both, so it cannot be correct on both. Error at most $1/3$ on each requires a query with probability at least $1/3$, and hence at least $1/3$ expected rounds. Taking the maximum of this constant bound and the preceding bound gives the required sum.

Total queries. For the upper bound, use $b = 1$ and $g = \min\{B, m - 1\}$ in Theorem D.1. The initial tests and binary edge searches use $\mathcal{O}(m \log(1/\varepsilon))$ queries. For the lower bound, unfold each batch into its individual binary answers. This gives a binary query tree with the same total query count and output as the original algorithm. The Q^{m-1} distinct acceptable sets in Theorem D.3 require distinct

correct outputs. Lemma D.2 therefore gives $\Omega((m-1)\log Q) = \Omega(m\log(1/\varepsilon))$ expected queries, even with error at most $1/3$. □

D.6 Multi-agent lower bound

Proof of Theorem 4.3. We use instances in which the first m agents determine a unique consensus lottery, while the additional agents must still be checked. The construction for the first m agents is from Choo et al. (2026, Lemma D.2). For each $b = (b_1, \dots, b_m) \in \mathbb{Z}^m_{>0}$ with $\sum_i b_i = Q$, give agent $i \in [m]$ utility one on s_i, utility zero elsewhere, and threshold $b_i\varepsilon$. Every additional agent has utility one on s_1, utility zero elsewhere, and threshold ε. The first m constraints require $x_i \geq b_i\varepsilon$ for all i. Since both sides sum to one, their only common acceptable lottery is $\mathbf{x}^{(b)} = (b_1\varepsilon, \dots, b_m\varepsilon)$. The additional agents accept it because $b_1 \geq 1$.

Our family consists of all these feasible instances and all instances obtained from them by changing one additional agent's threshold to one. Each changed agent accepts only $\mathbf{e}_1$. Since every coordinate of $\mathbf{x}^{(b)}$ is positive and $m \geq 2$, the changed instance is infeasible. All agents in both types of instances remain nontrivial and quantized.

There are $M_0 = \binom{Q-1}{m-1}$ feasible instances in the family, with distinct required outputs. In a batch, let q_i count the distinct coordinate values queried to agent $i \in [m]$. This agent's answers have at most $q_i + 1$ possibilities as b_i varies. On the feasible instances, queries to the additional agents have answers independent of b. Thus a batch of at most B queries has at most

$$\Phi_{m,B} = \max_{q_1,\dots,q_m \in \mathbb{Z}_{\geq 0}, q_1+\cdots+q_m \leq B} \prod_{i=1}^{m} (q_i + 1)$$

answer vectors. Lemma D.2 gives $\Omega(\log M_0 / \log \Phi_{m,B})$ expected rounds on some feasible instance, even with error at most $1/3$.

We also obtain a checking bound on every feasible instance in the family. Fix an additional agent $\ell > m$ and compare that instance with its infeasible variant in which only ℓ is changed. With the same internal randomness, the two executions are identical until ℓ is queried. If it is never queried, their common output cannot be correct on both. The sum of the two error probabilities is at most $2/3$, so ℓ is queried on the feasible instance with probability at least $1/3$. Summing over additional agents yields at least $(n-m)/3$ expected queries, and hence at least $(n-m)/(3B)$ expected rounds.

To simplify the first bound, use the assumption $m \leq Q^{1-\delta}$:

$$\log M_0 \geq (m-1)\log\frac{Q-1}{m-1} \geq (m-1)\log\frac{Q}{m} \geq \delta(m-1)\log Q = \Omega(m\log Q).$$

The first inequality follows from $\binom{a}{b} \geq (a/b)^b$ for integers $a \geq b \geq 1$. For $B \leq m-1$, the inequality $q_i + 1 \leq 2^{q_i}$ gives $\Phi_{m,B} \leq 2^B$, yielding $\Omega((m/B)\log Q)$ rounds. For $B \geq m-1$, the arithmetic–geometric mean inequality gives $\Phi_{m,B} \leq (1 + B/m)^m$, yielding $\Omega(\log Q / \log(1 + B/m))$ rounds. For sufficiently large Q, there are two feasible instances with different required outputs; the same no-query comparison as in the preceding proof gives an additional $\Omega(1)$ expected rounds. All these bounds use the same family. Their maximum is at least a constant multiple of their sum, which proves the theorem. □

D.7 Joint learning of a chosen set

Proof of Theorem 4.4. First choose the necessary edges for each agent, sharing the batch cap across all chosen agents. Query the km agent–pure-lottery pairs in $\lceil km/B \rceil$ rounds. For each agent separately, choose $\rho \in L_{\mathrm{rej}}$ and query $\mathbf{x}_{\rho,j,1-\varepsilon}$ for every $j \in L_{\mathrm{acc}}$. There are at most $k(m-1)$ such tests, taking at most $\lceil k(m-1)/B \rceil$ further rounds. These tests distinguish $\alpha_{\rho,j} < 1$ from $\alpha_{\rho,j} = 1$, exactly as in Theorem D.1. For an agent whose tests are all rejected, the reconstruction lemma already determines its acceptable set. For every other agent, the same edge choices leave at most $m-1$ thresholds in Γ^-_ε to identify. Thus there are at most $k(m-1)$ edge searches in total.

If $B \leq k(m-1)$, divide these searches into groups of at most B edges and run binary search on every edge of a group in parallel. Including the initial tests, the number of rounds is $\mathcal{O}((km/B)\log(1/\varepsilon))$. Since $B/(km) \leq 1$, this equals the stated asymptotic bound, using $\log(1 + B/(km)) = \Theta(B/(km))$.

If $B \geq k(m-1)$, search all remaining edges together, with at most $\lfloor B/[k(m-1)] \rfloor$ queries per edge per round. The initial tests take $\mathcal{O}(1)$ rounds, and Lemma 4.1 bounds the search rounds by

$$\mathcal{O}\left(1 + \frac{\log(1/\varepsilon)}{\log(1 + B/(km))}\right).$$

Every batch has at most B queries. In both cases, Lemma C.2 gives all k exact inequalities once the thresholds are known.

For the total query bound, use binary search on each edge for every B, grouping at most $\min\{B, k(m-1)\}$ searches together. There are $\mathcal{O}(km)$ initial queries and at most $k(m-1)$ searches, each using $\mathcal{O}(\log(1/\varepsilon))$ queries. The resulting total is $\mathcal{O}(km \log(1/\varepsilon))$. □

D.8 Weighted random sampling

For a set or multiset G of agents, write $\mathbf{x}(G)$ for the lottery returned by `Select` on their constraints, or NULL if their intersection is empty. Repeated copies of an agent impose the same constraint. The proof of Proposition C.3 implies the following fact: every G has a subset of at most m agents whose constraints alone give $\mathbf{x}(G)$. If $\mathbf{x}(G) \neq$ NULL, at most $m-1$ agents suffice.

The following is the sampling lemma of Gärtner & Welzl (2001, Lemma 1.1), applied to copies representing the integer weights used by Clarkson (1995).

Lemma D.4. *Give each agent a positive integer weight, let W be the total weight, and view the weights as copies in a multiset. Treat the copies as distinct when sampling. Choose an integer $1 \leq k < W$ and sample k copies uniformly without replacement; denote this sample by G. Then the expected total weight of agents that reject $\mathbf{x}(G)$ is at most $\frac{m(W-k)}{k+1}$. If $\mathbf{x}(G) =$ NULL, define this rejected weight to be zero.*

Proof. Count the sample as a set of distinct copies, even when several copies represent the same agent. For a set T of $k+1$ copies, count a copy $h \in T$ when $\mathbf{x}(T \setminus \{h\}) \neq$ NULL and its agent rejects that lottery. Choose at most m copies whose constraints determine $\mathbf{x}(T)$. If h is not among those copies, their constraints are still present in $T \setminus \{h\}$, so removing h does not change $\mathbf{x}(T)$. If this common result is a lottery, the agent of h accepts it; if it is NULL, h is not counted by definition. Thus at most m copies in T are counted.

Now count pairs (G, h), where G is a k-copy sample, h is outside G, and the agent of h rejects $\mathbf{x}(G) \neq$ NULL. For any such lottery, every copy of a rejecting agent lies outside G, since sampled agents accept $\mathbf{x}(G)$. Hence, for a fixed G, the number of these pairs equals the total rejected weight. Each $(k+1)$-copy set T contributes at most m pairs, by the preceding paragraph. Averaging over the $\binom{W}{k}$ equally likely samples gives

$$\mathbb{E}[\text{rejected weight}] \leq m \frac{\binom{W}{k+1}}{\binom{W}{k}} = \frac{m(W-k)}{k+1}.$$

Samples for which $\mathbf{x}(G) =$ NULL contribute zero, as required. □

Proof of Theorem 4.5. We implement the weighted-sampling method of Choo et al. (2026), following the weight analysis of Clarkson (1995), while counting both learning and checking rounds. If $n \leq \lceil 6m^2 \rceil$, learn all agents directly and apply `Select`; this meets the claimed bounds. Otherwise set $k = \lceil 6m^2 \rceil$ and initially give every agent weight one.

In each iteration, sample k copies uniformly without replacement from the current weighted multiset, and learn every distinct sampled agent. Apply `Select` to the sampled constraints. If it returns NULL, return NULL. Otherwise query all agents at the selected lottery, and let V be the set of rejecting agents. If $V = \varnothing$, return the lottery. If the total weight of V is at most $W/(3m)$, double the weight of every agent in V; otherwise leave all weights unchanged. Here W is the current total weight, so $W \geq n > k$ in every iteration.

Whenever the algorithm returns, its answer is correct: an infeasible sample makes the full instance infeasible, and a returned lottery has been accepted by every agent. To bound the number of iterations, condition on all previous samples and answers. Lemma D.4 bounds the expected rejected weight of

the new sample by $mW/(k+1) \le W/(6m)$. Markov's inequality implies that, with probability at least $1/2$, the algorithm either returns or changes the weights. This conditional bound holds after every preceding sequence of iterations.

By the fact above, fix a set $D^* \subseteq N$ of at most m agents with $\mathbf{x}(D^*) = \mathbf{x}(N)$. Every iteration that continues with a nonempty V has $V \cap D^* \neq \varnothing$. For a feasible instance, the selected lottery for the sample is lexicographically no smaller than $\mathbf{x}(N)$, since fewer constraints are imposed. If every agent in D^* accepted it, it would also be lexicographically no larger than $\mathbf{x}(D^*) = \mathbf{x}(N)$. It would equal $\mathbf{x}(N)$, contradicting $V \neq \varnothing$. For an infeasible instance, no lottery is accepted by all agents in D^*. If $D^* = \varnothing$, the feasible-case argument already shows that the algorithm returns in its first iteration.

After t weight-changing iterations, some agent in D^* has had its weight doubled at least t/m times. Thus $W \ge 2^{t/m}$. On the other hand, a weight change adds at most $W/(3m)$ to the current total, so

$$2^{t/m} \le W \le n\left(1 + \frac{1}{3m}\right)^t \le ne^{t/(3m)}.$$

Taking natural logarithms gives $t(\ln 2 - 1/3) \le m \ln n$. The number of weight changes is therefore $\mathcal{O}(m \log(n+1))$ on every execution. The conditional probability bound above implies that the expected number of iterations until the next weight change or return is at most two. Summing these conditional expectations over the bounded number of weight changes and one final return gives an expected total of $\mathcal{O}(m \log(n+1))$ iterations. In particular, the algorithm terminates with probability one.

An iteration uses at most R' rounds and Q' queries for sample learning, followed by at most $\lceil n/B \rceil$ rounds and n queries for checking. All batches respect the global cap B. Multiplying by the expected number of iterations proves both bounds. Reusing previously learned inequalities can only reduce these costs. The direct all-agent algorithm in Theorem 4.4 provides the stated alternative. □

D.9 Two alternatives

By nontriviality, one query at $\mathbf{e}_1$ determines whether an agent imposes a lower bound or an upper bound on β in $\mathbf{x}_{1,2,\beta}$. Lower endpoints lie in Γ_ε, and upper endpoints lie in $\{0\} \cup \Gamma_\varepsilon^-$. A nonempty consensus interval contains a value in $\{0\} \cup \Gamma_\varepsilon$: its left endpoint is either 0 or one of the lower endpoints. The number of candidate values is $|\Gamma_\varepsilon| + 1 = \Theta(\varepsilon^{-2})$.

The algorithms below maintain a consecutive set $C \subseteq \{0\} \cup \Gamma_\varepsilon$ and a set of agents still to be checked. Every feasible candidate remains in C, and every removed agent accepts every candidate in C. These properties remain valid if the problem has already been restricted to an interval with endpoints in $\{0\} \cup \Gamma_\varepsilon$.

D.9.1 Direct feasibility search

We first analyze Algorithm 1. At a median candidate β, a lower-bound rejection excludes β and every smaller candidate; an upper-bound rejection excludes β and every larger candidate. An earlier accepting agent of the rejecting agent's direction accepts every candidate on the retained side, so it can be removed. Thus every feasible candidate remains in C, and every removed agent accepts every candidate in C. An all-accepting scan returns a consensus lottery. If C becomes empty, no consensus exists, because every nonempty feasible interval contains an initial candidate in $\{0\} \cup \Gamma_\varepsilon$.

If a rejection removes d earlier accepting agents of its direction, the alternating order has queried at most $d+1$ agents of the other direction. The scan uses at most $2(d+1)$ queries. This remains true when one list is exhausted, since the rejecting agent's own list was not exhausted before its query. Each agent is removed at most once, and each rejection reduces the candidate count to at most half its previous value. Hence there are at most $\lceil \log_2(|\Gamma_\varepsilon| + 2) \rceil$ rejections. The final all-accepting scan, if any, queries each remaining agent once. The removed agents and the agents in that final scan are disjoint, so their combined contributions to the estimates above total at most $2n$. Including the n direction queries and two queries per rejection gives at most

$$3n + 2\lceil \log_2(|\Gamma_\varepsilon| + 2) \rceil = \mathcal{O}(n + \log(1/\varepsilon))$$

queries. This algorithm is valid for every batch cap, since it asks one query at a time. The following extension reduces the number of rounds.

Lemma D.5. *Suppose a agents have known inequality directions and the properties above hold for a nonempty candidate set C. For integers $b, g \geq 1$ with $bg \leq B$, consensus within C can be solved deterministically in $\mathcal{O}(\frac{a}{g} + 1 + \frac{\log(|C|+1)}{\log(b+1)})$ rounds.*

Proof. We test several candidates at once, but process agents in alternating directions as in the one-query search. A reduction of the candidate set will also let us remove previously queried agents. The two kinds of progress account for the two terms in the round bound. If no agents remain, return any candidate in C.

Set $t = \min\{b, |C|\}$ and choose t candidate values $z_1 < \cdots < z_t$. Choose their indices in C to be $\lfloor j(|C|+1)/(t+1) \rfloor$ for $j \in [t]$. Every open gap between consecutive values z_j, including the two outer gaps, then contains at most $\lfloor |C|/(t+1) \rfloor$ candidates.

Scan the agents in alternating lower- and upper-bound order, continuing with the other direction if one list is exhausted. In each round, query the next at most g agents at all of $z_1, \ldots, z_t$, then process their answer vectors in the scan order. This batch is chosen before its answers are known and has at most $gt \leq B$ queries. Among the processed agents, let j be the largest index such that z_j is rejected by a lower-bound agent, or zero if none is rejected. Let j' be the smallest index such that $z_{j'}$ is rejected by an upper-bound agent, or $t+1$ if none is rejected. The values accepted by every processed agent are exactly $z_{j+1}, \ldots, z_{j'-1}$.

If $j \geq j'$, the processed constraints require a value strictly above z_j and strictly below $z_{j'}$, which is impossible; return NULL. If $j' = j+1$, a feasible candidate can lie only in the open gap between z_j and $z_{j'}$, with the same interpretation for an outer gap. Restrict C to this gap, returning NULL if it is empty. If neither case occurs by the end of the scan, return any of $z_{j+1}, \ldots, z_{j'-1}$; every remaining agent accepts it.

Suppose a lower-bound agent first makes $j' = j+1$. Before its answer was processed, the largest lower rejection index was at most $j-1$. Every earlier lower-bound agent therefore accepts z_j and all candidates in the gap to its right, so remove all those earlier agents. The upper-bound case is symmetric. Keep the agent that caused the restriction. Ignore the answers of any later agents in the same batch and keep them for the next scan. Thus no feasible candidate is removed, and every removed agent accepts all candidates in the new C. These are precisely the required properties, so repeating the scan is valid.

If a restriction removes d agents, the alternating order processes at most $2(d+1)$ agents up to the answer causing that restriction. The scan therefore takes at most $\lceil 2(d+1)/g \rceil \leq 2d/g + 3$ rounds, including any unused answers from its last batch. The sum of these removal counts is at most a. Each restriction leaves at most $\lfloor |C|/(b+1) \rfloor$ candidates; when $|C| < b$, all candidates were queried and the chosen gap is empty. Hence there are $\mathcal{O}(1 + \log(|C|+1)/\log(b+1))$ such restrictions, where $|C|$ in this bound is the initial size. A final scan that returns a lottery or finds inconsistent constraints uses at most $\lceil a/g \rceil$ rounds. Adding these costs proves the claimed bound and also shows termination. □

D.9.2 Reducing the number of agents by sampling

The following bound applies the estimate for the smallest sampled value from Gärtner & Welzl (2001, Section 2) separately to lower and upper endpoints. The endpoint tests ensure that each removed agent accepts the entire remaining interval.

Lemma D.6. *Suppose $a \geq k \geq 1$ agents with known inequality directions remain. Assume every consensus value lies in a known interval $[\ell, u]$ and every previously removed agent accepts that interval. Choose k remaining agents uniformly without replacement, learn their endpoints, and intersect their acceptable intervals with $[\ell, u]$. If the intersection is empty, return* NULL*. Otherwise, test each remaining lower-bound agent at the new left endpoint and each remaining upper-bound agent at the new right endpoint, and remove agents that accept. The expected number of agents left is at most $2(a-k)/(k+1)$, counting it as zero if the algorithm terminates.*

Proof. Every consensus value satisfies the sampled constraints and the previous interval restriction, so it remains in the updated interval. If that interval is empty, returning NULL is therefore correct.

Otherwise, a lower-bound agent accepting its left endpoint accepts the whole interval, and an upper-bound agent accepting its right endpoint does too. Thus removing these agents preserves the stated properties.

To bound the number left, order the a agents by putting lower-bound agents first in decreasing order of endpoint, with arbitrary tie-breaking, and then all upper-bound agents. Every lower-bound agent left after the tests precedes the first sampled agent in this order. If a lower-bound agent was sampled, this follows because a remaining lower endpoint must be strictly larger than every sampled lower endpoint. If none was sampled, all lower-bound agents precede the first sampled upper-bound agent.

For a uniform k-subset of a ordered positions, the expected number of positions before the first sampled one is $(a-k)/(k+1)$. Indeed, the $k+1$ gaps of unsampled positions have the same expected size and their sizes sum to $a-k$. This bounds the expected number of remaining lower-bound agents. Apply the same argument with upper-bound agents first, in increasing order of endpoint, to bound their expected number by $(a-k)/(k+1)$. Adding gives $2(a-k)/(k+1)$. Ties, the pre-existing interval restriction, and termination with an empty intersection can only reduce the number left. □

Proof of Theorem 4.6. **Randomized round upper bound.** First determine all inequality directions in $\lceil n/B \rceil$ rounds. Set $k = \lceil \sqrt{B} \rceil$ and $b = \lfloor B/k \rfloor$. These are positive integers and $kb \le B$. Start with $[\ell, u] = [0, 1]$ and perform at most two sampling steps from Lemma D.6, using fresh uniform samples. In each step, learn the k sampled endpoints in parallel, with at most b queries per sampled agent in a round. If at most k agents remain, learn all their endpoints with the same allocation and solve the restricted problem immediately.

Every endpoint has at most $|\Gamma_\varepsilon| + 1$ candidates; use the reverse candidate order for upper-bound endpoints. Lemma 4.1 therefore bounds the learning rounds of each step by

$$\mathcal{O}\left(1 + \frac{\log(1/\varepsilon)}{\log(b+1)}\right) = \mathcal{O}\left(1 + \frac{\log(1/\varepsilon)}{\log(B+1)}\right).$$

The equality uses $\log(b+1) = \Theta(\log(B+1))$ uniformly for integer $B \ge 1$. Indeed, $b \ge \sqrt{B}/4$, which gives this comparison together with $1 \le b \le B$. Each step's endpoint tests take at most $\lceil n/B \rceil$ more rounds.

Condition on the samples and answers before a step. If a agents remain, Lemma D.6 bounds the expected number after the step by $2a/(k+1)$. Count this number as zero if the algorithm has already returned. Applying conditional expectation twice, after at most two steps the expected number remaining is at most $\frac{4n}{(k+1)^2} \le \frac{4n}{B}$. Use Lemma D.5 on those remaining agents, with the lemma's parameters $g = 1$ and $b = B$, and with $C = (\{0\} \cup \Gamma_\varepsilon) \cap [\ell, u]$. The interval endpoints belong to $\{0\} \cup \Gamma_\varepsilon$, so C is nonempty whenever the interval is nonempty. The sampling steps preserve every consensus value and remove only agents accepting the full current interval, so this application is valid. Its expected round count is $\mathcal{O}(1 + n/B + \log(1/\varepsilon)/\log(B+1))$. Adding the direction queries and the two sampling steps gives the claimed upper bound. Every batch respects the global cap, and every choice of the samples leads to a correct answer after finitely many rounds.

Randomized lower bounds and total queries. Use the family from Theorem C.6 with two opposite inequalities at a common unknown $\alpha \in \Gamma_\varepsilon^-$, and $n-2$ additional agents requiring $\beta \ge \varepsilon$. The unique consensus lottery is $\mathbf{x}_{1,2,\alpha}$. A batch of at most B queries gives at most $2B+1$ answer vectors, including equality cases. When $|\Gamma_\varepsilon^-| \ge 2$, Lemma D.2 gives $\Omega(\log(1/\varepsilon)/\log(B+1))$ expected rounds. Unfolding each batch into its binary answers gives $\Omega(\log(1/\varepsilon))$ expected total queries as well.

The checking argument in Theorem C.6 gives $\Omega(n)$ expected queries, hence $\Omega(n/B)$ expected rounds. There is also an $\Omega(1)$ round bound, since two instances with different required outputs cannot both be answered correctly without making a query with probability at least $1/3$. These bounds also cover $|\Gamma_\varepsilon^-| = 1$, when $\varepsilon = 1/2$ and the precision term is bounded by a constant. Taking maxima, and then using that a maximum is at least a constant multiple of the sum of these terms, proves the randomized round lower bound and the total query lower bound $\Omega(n + \log(1/\varepsilon))$. The direct one-query search analyzed above achieves the total query bound deterministically for every B.

Deterministic round upper bound. After determining the directions, apply Lemma D.5 with any integer $1 \le b \le B$ and $g = \lfloor B/b \rfloor$. Since $g \ge B/(2b)$, the total number of rounds is

$$\mathcal{O}\left(1 + \frac{nb}{B} + \frac{\log(1/\varepsilon)}{\log(b+1)}\right).$$

Choose $b = \max\{1, \lfloor\sqrt{\min\{B,\, B\log(1/\varepsilon)/n\}}\rfloor\}$. If $B\log(1/\varepsilon)/n < 1$, then $b = 1$ and $\log(1/\varepsilon) < n/B$, giving $\mathcal{O}(1 + n/B)$ rounds. Otherwise,

$$\log(b+1) = \Theta\left(\log(2 + \min\{B, B\log(1/\varepsilon)/n\})\right),$$

and $b^2 \le B\log(1/\varepsilon)/n$ gives

$$\frac{nb}{B} \le \frac{\log(1/\varepsilon)}{b} \le \frac{\log(1/\varepsilon)}{\log(b+1)}.$$

Substituting these estimates proves the deterministic upper bound.

Deterministic round lower bound. Let R be the worst-case number of rounds, so the total queries are at most BR. Apply Lemma C.7, using $p = \lfloor n/2 \rfloor \ge n/3$:

$$\log(|\Gamma_\varepsilon^-|/2) \le R\log\left(1 + \frac{6BR}{n}\right).$$

For sufficiently small ε, the left side is $\Omega(\log(1/\varepsilon))$. If $R \ge \log(1/\varepsilon)$, the bound below is immediate. Otherwise substitute $\log(1/\varepsilon)$ for R inside the logarithm. In either case,

$$R = \Omega\left(\frac{\log(1/\varepsilon)}{\log(2 + B\log(1/\varepsilon)/n)}\right).$$

The randomized lower bounds already give $R = \Omega(1 + n/B)$ and $R = \Omega(\log(1/\varepsilon)/\log(B+1))$ for deterministic algorithms. The larger of the two logarithmic ratios uses the smaller denominator, which is within a constant factor of $\log(2 + \min\{B, B\log(1/\varepsilon)/n\})$. Combining the three bounds gives the stated deterministic expression. For bounded $1/\varepsilon$, the bound $\Omega(1 + n/B)$ suffices. All constants in this theorem are absolute. □