SELECTIVITY ESTIMATION IN DATABASES VIA ONLINE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning-based approaches for selectivity estimation in databases have gained significant traction in recent years. In this paper, we introduce a new framework for this problem that formulates selectivity estimation as an *online learning* problem. Unlike previous approaches, our framework does not require the underlying database or query distribution to remain fixed and naturally accommodates evolving data and query workloads. Within this framework, we derive tight regret bounds for a broad class of queries under standard loss functions. Furthermore, we complement our theoretical analysis with an extensive empirical evaluation on real-world datasets. Our theoretical and empirical results establish online learning as a principled framework for accurate and adaptive selectivity estimation in dynamic database environments.

1 INTRODUCTION

A large number of learning-based methods for fundamental database operations have emerged in recent years in both theory and practice (Kraska et al., 2018; Marcus et al., 2019; Kristo et al., 2020; Hu et al., 2022). A prominent example is the core problem of *selectivity estimation*: predicting the fraction of rows in a database table that satisfy a specific query predicate or condition, which is a key input to query optimizers choosing between candidate query execution plans, a central task in database management systems (Dutt et al., 2019; Yang et al., 2020; Park et al., 2020; Negi et al., 2023; Zeng et al., 2024; Zhu et al., 2025). By leveraging machine learning models to capture complex data distributions and query patterns, these methods have potential for significantly improving accuracy compared to traditional approaches for selectivity estimation (Wu et al., 2025; Zeighami & Shahabi, 2024a;b). However, in spite of significant empirical success, theoretical understanding of learning-based approaches for selectivity estimation remains limited. Prior theoretical studies, such as Hu et al. (2022), focus on static settings where the underlying database and query workloads remain fixed over time, and the learning algorithm can train on sample queries from the same workload that it is asked to estimate selectivity for. In contrast, real-world database environments are frequently *dynamic*: data is continuously inserted, updated, and deleted, and query patterns may shift unpredictably without following any stationary distribution. Under such dynamics, static models trained on historical data can become stale, leading to degraded accuracy or necessitating expensive retraining (Wang et al., 2021). While practical systems attempt to handle these dynamics through robust training and lightweight adaptation (Li et al., 2022; Negi et al., 2023; Zhu et al., 2025), our goal in this work is to provide a rigorous theoretical framework that accommodates these fully dynamic, distribution-free settings, thereby enabling principled solutions to this important problem.

Our key novelty in terms of capturing a dynamic environment is to formulate selectivity estimation as an *online learning* problem. This allows both databases and incoming queries to change arbitrarily over time, and does not impose any distributional assumptions on either. The online learning framework naturally mirrors the sequential nature of query processing, where the learner operates in sequential rounds and prediction must be made before true selectivity is revealed. Concretely, in each round: (1) a query arrives; (2) the learner predicts its selectivity; (3) the true selectivity is revealed after query execution; and (4) the learner incurs a loss based on its prediction error. The learner uses only past query–selectivity pairs to predict the selectivity of each incoming query, without direct access to the underlying database. This models learning from query execution feedback without additional database scans for training. As is standard in online learning, the goal of an algorithm is to minimize *regret*, i.e., the cumulative loss incurred by the algorithm compared to minimum

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.

loss incurred by a fixed strategy in hindsight. Within this framework, we give tight regret bounds for the entire class of linear queries under standard loss functions.

Our Contributions. We give an outline of our main contributions below.

- Our first contribution is a novel online-learning based model for selectivity estimation. This framework allows us to derive rigorous bounds for learned selectivity estimation in a distribution-free manner in dynamic environments. This model is presented in Section 2.
- In our technical results, we distinguish between static and dynamic databases, and consider two standard loss functions—squared and absolute loss—for both settings. For linear queries, we derive (nearly) matching upper and lower bounds on the regret of the online learner in all four resulting categories. These results are given in Theorem 2.1 and Theorem 2.2 respectively in Section 2, with proof sketches in Sections 3 to 5 (detailed proofs in the appendix).
- To complement our theoretical results, we empirically evaluate our online-learning based selectivity estimation algorithms on real-world datasets under varying query workloads and dataset settings. Our algorithms achieve empirical regret well below the corresponding theoretical upper bounds and estimation accuracy competitive with the evaluated baselines. Section 6 gives an overview of our main experimental findings; additional experiments are in the appendix.

Related work. Existing theoretical analysis of learned selectivity estimation and database operations establish sample-complexity and generalization guarantees under distributional assumptions (Hu et al., 2022; Zeighami & Shahabi, 2024a;b; Zhu et al., 2025). In contrast, we establish distribution-free regret guarantees for online selectivity estimation under arbitrary database and query sequences. From a more abstract learning-theoretic perspective, Anderson & Benedikt (2025) study how learnability transfers from base function classes to derived statistical function classes in both PAC and online settings. Our work complements these general learnability results by characterizing achievable regret for a concrete selectivity estimation problem. Our formulation also falls within online convex optimization over the probability simplex (Shalev-Shwartz, 2012; Hazan, 2016). We exploit the specific structure of selectivity estimation to establish tight regret bounds. A detailed discussion of related work, including connections to convex body chasing, is in Appendix A.

2 Model and Results

2.1 Online-learning based framework for selectivity estimation

Selectivity function. The underlying database is represented by a histogram on a discrete domain of size k, i.e., a vector $\mathbf{w} \in \mathbb{R}^k_{\geq 0}$, where the i-th coordinate w_i represents the relative frequency of the i-th element e_i. (This can be achieved even on continuous domains by partitioning into k intervals of appropriate granularity.) The set of all valid database vectors forms the k-dimensional simplex

$$\Delta_k := \left\{ \mathbf{w} \in \mathbb{R}^k \;\middle|\; \sum_{i=1}^{k} w_i = 1,\ w_i \geq 0 \ \forall i \in [k] \right\}.$$

We study *linear queries*, represented by k-dimensional vectors $\mathbf{v} \in [0,1]^k$. Each component $\mathbf{v}_i$ specifies the weight or fractional coverage of data at the i-th element e_i. Selectivity of a query $\mathbf{v}$ is defined by the weighted sum

$$\sigma(\mathbf{v}, \mathbf{w}) = \langle \mathbf{v}, \mathbf{w} \rangle = \sum_{i=1}^{k} v_i w_i.$$

We note that linear queries include many commonly studied query classes including *point* queries, which select a single domain element ($\mathbf{v} = \mathbf{e}_i$), *range* queries, which select a contiguous interval of an ordered domain, *subset queries*, which select an arbitrary subset of the domain, and so on. (In some cases, we give tighter regret bounds for special cases of linear queries in Appendix B, but we only discuss linear queries in the main paper to keep it coherent.)

Online learning steps. The selectivity estimation problem asks a learner to predict the selectivities of a sequence of T queries from a given query class $\mathcal{Q}$ (linear queries in this paper) based only on prior history. Let $\mathbf{v}_t$ denote the t-th query and σ_t be its true selectivity. In round $t = 1, \ldots, T$, we have the following steps:

1. A query vector $\mathbf{v}_t \in \mathcal{Q}$ is presented to the learner (we call the learner algorithm $\mathcal{A}$).
2. The learner outputs a selectivity prediction $\hat{\sigma}_t \in [0,1]$ based on $\mathcal{H}_t := (\mathbf{v}_1, \sigma_1, \mathbf{v}_2, \sigma_2, \ldots, \mathbf{v}_{t-1}, \sigma_{t-1}, \mathbf{v}_t)$, i.e., all the previous queries-selectivity pairs as well as the new query (but not its selectivity).
3. Next, the true selectivity σ_t of query $\mathbf{v}_t$ is revealed. Note that σ_t corresponds to a (possibly time-varying) unknown database vector $\mathbf{w}_t \in \Delta_k$, i.e., $\sigma_t = \sigma(\mathbf{v}_t, \mathbf{w}_t) = \langle \mathbf{v}_t, \mathbf{w}_t \rangle$. The learner observes only σ_t, but not the underlying $\mathbf{w}_t$.
4. The learner incurs a loss based on the difference between the true and predicted selectivities: $\text{Loss}_t := \text{Loss}(\sigma_t, \hat{\sigma}_t)$.

Loss functions and Regret. We consider two standard loss functions: (i) **squared loss**: $\text{Loss}_t = (\sigma_t - \hat{\sigma}_t)^2$; and (ii) **absolute loss**: $\text{Loss}_t = |\sigma_t - \hat{\sigma}_t|$, and measure the performance of an online learning algorithm $\mathcal{A}$ using *regret* against the best fixed comparator, denoted $\text{Regret}_T(\mathcal{A})$.

A fixed comparator $\mathbf{w} \in \Delta_k$ predicts the selectivity in round t as $\sigma(\mathbf{v}_t, \mathbf{w}) = \langle \mathbf{v}_t, \mathbf{w} \rangle$ and its loss in round-t is given by $\text{Loss}_t(\mathbf{w}) := \text{Loss}(\sigma_t, \langle \mathbf{v}_t, \mathbf{w} \rangle)$. Therefore, regret of algorithm $\mathcal{A}$ is given by:

$$\text{Regret}_T(\mathcal{A}) = \sum_{t=1}^{T} \text{Loss}_t - \min_{\mathbf{w} \in \Delta_k} \sum_{t=1}^{T} \text{Loss}_t(\mathbf{w}). \tag{1}$$

Our goal is to design algorithms that minimize regret under squared and absolute loss functions.

Static vs. dynamic databases. We analyze the framework in two database settings. In **static databases**, there is a fixed but unknown database vector $\mathbf{w}^* \in \Delta_k$ such that $\mathbf{w}_t = \mathbf{w}^*$ in every round, while queries may still vary arbitrarily. This models environments such as analytical warehouses, where the underlying data is updated infrequently in batches, yet the query workload may shift over time as analysts explore different aspects of the data. In contrast, in **dynamic databases**, the database state $\mathbf{w}_t$ (as well as the queries) can change arbitrarily across rounds. This models transactional or production systems with continuous insertions, updates, and deletions, where the learner must simultaneously adapt to an evolving database and query workload.

As one might expect, we observe a qualitative difference in the performance of online learning algorithms in the two settings. Before stating our results formally, we make an important observation about static databases that is key to distinguishing it from the dynamic case. Note that for a static database, the optimal comparator $\mathbf{w}^*$ incurs zero loss, which means that the regret of the algorithm is given by its cumulative loss. In essence, the problem reduces to identifying $\mathbf{w}^*$ from the revealed query-selectivity pairs. Each revealed $\sigma_t = \langle \mathbf{v}_t, \mathbf{w}^* \rangle$ is a linear constraint on $\mathbf{w}^*$; since the selectivity of a query that is a linear combination of previous queries can be estimated exactly and there are only $k-1$ linearly independent queries each incurring at most unit loss, the total loss is no more than $k-1$. This gives a baseline for the static setting, as summarized in the following observation:

Observation 1. *For a static database, the regret of any algorithm $\mathcal{A}$ equals its cumulative loss: $\text{Regret}_T(\mathcal{A}) = \sum_{t=1}^{T} \text{Loss}_t$. Moreover, there exists an online algorithm whose regret is at most $k-1$ under either squared or absolute loss, uniformly over all horizons T. Thus, the regret upper bound is independent of T.*

2.2 Our Results

We are now ready to formally state our theoretical results. For static databases, we propose the Sequential Maximum Entropy (SeqMaxEnt) algorithm and establish regret bounds substantially sharper than the $k-1$ baseline of Observation 1. We prove (nearly) matching lower bounds under both loss functions. Theorem 2.1 summarizes these results.

Theorem 2.1 (**Static databases**). *The* SeqMaxEnt *algorithm (Algorithm 1) achieves $O(\log k)$ regret under squared loss, and $O(\sqrt{k \log k})$ regret under absolute loss.*

Further, any (possibly randomized) algorithm incurs expected regret $\Omega(\log k)$ under squared loss, and $\Omega(\sqrt{k/\log k})$ under absolute loss in the worst case.

Therefore, for selectivity estimation on a static database with support size k, the optimal regret is $\Theta(\log k)$ under squared loss and $\tilde{\Theta}(\sqrt{k})$[1] under absolute loss.

[1] $\tilde{\Theta}(\cdot)$ is an asymptotic notation that ignores poly-logarithmic factors.

When the database is changing over time, the regret necessarily depends on T in the worst case. We therefore treat T, rather than k, as the dominant parameter in the dynamic database setting. We show that well-known algorithms for online convex optimization, when applied to selectivity estimation, achieve regret bounds with optimal dependence on T. In particular, we prove that (1) the *Exponentially Weighted Online Optimization (EWOO) algorithm (Hazan et al., 2007)* achieves $O(k \log T)$ regret under squared loss, and (2) the Follow-the-Regularized-Leader (FTRL) algorithm (Shalev-Shwartz, 2012) with negative-entropy regularization achieves $O(\sqrt{T \log k})$ regret under absolute loss (Appendices D and E). Somewhat surprisingly, we give matching lower bounds for both cases that establish optimality of these generic online learning algorithms for our specific selectivity estimation task in the dynamic setting. These are summarized in Theorem 2.2.

Theorem 2.2 (**Dynamic databases**). *Any (possibly randomized) algorithm incurs expected regret $\Omega(\log T)$ under squared loss, and $\Omega(\sqrt{T \log k})$ under absolute loss in the worst case.*

Therefore, for selectivity estimation on a dynamic database with support size k over T rounds, the optimal regret is $\Theta(\log T)$ in T under squared loss and $\Theta(\sqrt{T \log k})$ under absolute loss.

Interestingly, for both static and dynamic databases, the upper and lower bounds establish a polynomial separation between the two loss functions in the dominant parameter (k and T respectively). Thus, the choice of loss function significantly affects the achievable regret guarantees.

Technical highlights. To highlight some of our main technical ideas in this paper, we select three key results from the above two theorems and give an outline of the main ideas behind them.

1. **The SEQMAXENT algorithm: squared loss in Theorem 2.1 (see Section 3).** The SEQMAXENT algorithm (Algorithm 1) predicts using the *maximum-entropy* database vector consistent with all past query–selectivity observations. The key observation is that a large squared prediction error forces a substantial decrease in the entropy of this estimate. We establish this relation by bounding the squared loss by the incremental KL divergence between successive estimates, and then bounding this divergence by the entropy decrease. Thus the cumulative squared loss is bounded by the total entropy decrease from the initial uniform estimate to the final estimate, yielding $O(\log k)$ regret, an exponential improvement over $k - 1$ given by Observation 1.
2. **Lower bound under absolute loss in Theorem 2.1 (see Section 4).** The challenge is to force repeated prediction errors while keeping the underlying database fixed. We use the nonconstant rows of a normalized Hadamard matrix (Horadam, 2012) to encode $\Theta(k)$ independent random signs into a single perturbation of the uniform database. By orthogonality, each subset query isolates one previously unobserved sign, forcing $\Omega(\delta)$ expected absolute loss, where δ controls the perturbation scale. Larger perturbations increase this loss but also increase the probability of a negative database coordinate. Choosing $\delta = \Theta(1/\sqrt{k \log k})$ makes this probability small enough that replacing invalid candidates with a valid database preserves the $\Omega(\delta)$ lower bound. Summing over $\Theta(k)$ rounds yields $\Omega(\sqrt{k/\log k})$ expected regret.
3. **Lower bound under squared loss in Theorem 2.2 (see Section 5).** We obtain the lower bound by reducing the analysis to estimating an unknown Bernoulli mean. The adversary draws $\mu \sim \mathrm{Uniform}[0, 1]$ once and, conditional on μ, generates the selectivities as independent $\mathrm{Bernoulli}(\mu)$ samples. In this case, the expected regret in each round t is lower-bounded by the posterior variance of μ, which follows a Beta distribution with expected variance $\Theta(1/t)$. Thus, the regret lower bound becomes the harmonic sum over the T rounds, i.e., $\Omega(\log T)$.

Organization. The rest of the paper is organized as follows. Sections 3 to 5 give more details of the three results outlined above. The remaining results and proof details establishing Theorems 2.1 and 2.2 are presented in the appendix. Section 6 gives an overview of our main experimental findings; additional experiments and details of the experimental setup appear in the appendix.

3 SEQUENTIAL MAXIMUM ENTROPY ALGORITHM FOR STATIC DATABASES

We present the SEQMAXENT algorithm (Algorithm 1) that achieves optimal regret for both squared and absolute loss in static databases. Recall from Section 2 that the valid database vectors correspond to a simplex Δ_k, where k is the size of the domain. The SEQMAXENT algorithm maintains two key variables at all times t: a feasible region $\mathcal{C}_t$ in this simplex, and within it, a specific vector called the *database estimate* $\hat{\mathbf{w}}_t$. $\mathcal{C}_t$ is the set of database vectors consistent with all query-selectivity pairs observed before round t, and $\hat{\mathbf{w}}_t$ is the vector that maximizes entropy in $\mathcal{C}_t$ (intuitively, the "most

uniform" vector consistent with prior observations). In Line 1, the algorithm initializes $\mathcal{C}_1 = \Delta_k$ and $\hat{\mathbf{w}}_1 = (1/k, \ldots, 1/k)$, the maximum-entropy vector over $\mathcal{C}_1$. In each round t, upon receiving query $\mathbf{v}_t$, it predicts $\hat{\sigma}_t = \langle \mathbf{v}_t, \hat{\mathbf{w}}_t \rangle$. After observing σ_t, it refines the feasible region to $\mathcal{C}_{t+1} = \mathcal{C}_t \cap \{\mathbf{w} \in \Delta_k : \langle \mathbf{v}_t, \mathbf{w} \rangle = \sigma_t\}$ and updates $\hat{\mathbf{w}}_{t+1}$ to the maximum-entropy distribution over $\mathcal{C}_{t+1}$.

The update in Line 6 is a standard convex program, for which an ε-accurate solution can be computed in time polynomial in k, t, and $\log(1/\varepsilon)$ by standard convex-optimization methods (e.g., interior-point methods) (Boyd & Vandenberghe, 2004). The regret analysis below is for an exact optimizer.

Algorithm 1 Sequential Maximum Entropy (SEQMAXENT)

1: **Initialize:** Feasible region $\mathcal{C}_1 \leftarrow \Delta_k$, $\hat{\mathbf{w}}_1 \leftarrow (1/k, \ldots, 1/k)$
2: **for** $t = 1, 2, \ldots, T$ **do**
3: **Predict:** Receive query $\mathbf{v}_t$ and make a prediction on the selectivity $\hat{\sigma}_t \leftarrow \langle \mathbf{v}_t, \hat{\mathbf{w}}_t \rangle$
4: **Observe:** Receive true selectivity σ_t
5: **Update Feasible Region:** $\mathcal{C}_{t+1} \leftarrow \mathcal{C}_t \cap \{\hat{\mathbf{w}} \in \Delta_k \mid \langle \mathbf{v}_t, \hat{\mathbf{w}} \rangle = \sigma_t\}$
6: **Update Database Estimate:** Compute $\hat{\mathbf{w}}_{t+1}$ by maximizing entropy over $\mathcal{C}_{t+1}$:

$$\hat{\mathbf{w}}_{t+1} \leftarrow \arg\max_{\mathbf{w} \in \mathcal{C}_{t+1}} \left(-\sum_{i=1}^{k} w_i \ln(w_i) \right)$$

Proposition 3.1. *Under squared loss in the static database setting, the* SEQMAXENT *algorithm has regret at most* $2 \ln k$.

Our analysis uses the *Kullback–Leibler (KL) divergence* defined as follows: for $\mathbf{p}, \mathbf{q} \in \Delta_k$, $D_{\mathrm{KL}}(\mathbf{p} \| \mathbf{q}) = \sum_{i=1}^{k} p_i \ln \frac{p_i}{q_i}$, with the convention $0 \ln 0 = 0$. It is non-negative and satisfies $D_{\mathrm{KL}}(\mathbf{p} \| \mathbf{q}) = 0$ iff $\mathbf{p} = \mathbf{q}$ (Cover, 1999). Since $\hat{\mathbf{w}}_1$ is the uniform distribution $(1/k, \ldots, 1/k)$, for any $\mathbf{w} \in \Delta_k$ we have: $D_{\mathrm{KL}}(\mathbf{w} \| \hat{\mathbf{w}}_1) = \sum_{i=1}^{k} w_i \ln(k w_i) \le \ln k$, since $\sum_i w_i = 1$ and $w_i \ln w_i \le 0$ for $w_i \in [0, 1]$.

Lemma 3.2 bounds each round's squared loss by the KL divergence between successive database estimates. It uses that $\hat{\mathbf{w}}_{t+1}$ is consistent with the new observation: the prediction error then equals $\langle \mathbf{v}_t, \hat{\mathbf{w}}_{t+1} - \hat{\mathbf{w}}_t \rangle$, which Hölder's and Pinsker's inequalities bound by the KL divergence.

Lemma 3.2. *For any round* $t \in \{1, 2, \ldots, T\}$*, the squared loss incurred by the* SEQMAXENT *algorithm satisfies* $\mathrm{Loss}_t \le 2\, D_{KL}(\hat{\mathbf{w}}_{t+1} \| \hat{\mathbf{w}}_t)$.

Lemma 3.3 bounds the increase in that KL divergence by the KL divergence between the uniform vector and the current database estimate by using the property of our maximum-entropy choice.

Lemma 3.3. *For every round* t, $D_{KL}(\hat{\mathbf{w}}_{t+1} \| \hat{\mathbf{w}}_t) \le D_{KL}(\hat{\mathbf{w}}_{t+1} \| \hat{\mathbf{w}}_1) - D_{KL}(\hat{\mathbf{w}}_t \| \hat{\mathbf{w}}_1)$.

By Observation 1, regret equals cumulative loss in the static database setting. Combining Lemmas 3.2 and 3.3 and telescoping,

$$\mathrm{Regret}_T = \sum_{t=1}^{T} \mathrm{Loss}_t \le 2 \sum_{t=1}^{T} D_{\mathrm{KL}}(\hat{\mathbf{w}}_{t+1} \| \hat{\mathbf{w}}_t) \le 2 D_{\mathrm{KL}}(\hat{\mathbf{w}}_{T+1} \| \hat{\mathbf{w}}_1) \le 2 \ln k.$$

This establishes Proposition 3.1, which is the squared-loss upper bound in Theorem 2.1; the proofs of Lemmas 3.2 and 3.3 are given in Appendix C.2.

Upper bound under absolute loss. The Cauchy–Schwarz inequality relates cumulative absolute and squared loss over T rounds: $\sum_{t=1}^{T} |\sigma_t - \hat{\sigma}_t| \le \sqrt{T \sum_{t=1}^{T} (\sigma_t - \hat{\sigma}_t)^2}$. The SEQMAXENT algorithm incurs nonzero loss in at most $k - 1$ rounds. Applying this inequality to those rounds and using the squared-loss bound in Proposition 3.1 gives cumulative absolute loss, and hence regret, at most $\sqrt{2(k-1) \ln k} = O(\sqrt{k \log k})$. The proof is given in Appendix C.3.

4 LOWER BOUNDS UNDER ABSOLUTE LOSS FOR STATIC DATABASES

To prove the lower regret bound of $\Omega(\sqrt{k / \log k})$ for static databases under absolute loss in Theorem 2.1, we construct an instance using a *Hadamard matrix*.

Definition 4.1 (Normalized Hadamard Matrix, see e.g., Horadam (2012))**.** A square matrix $\mathbf{H} \in \{-1,+1\}^{d\times d}$ is a *normalized Hadamard matrix* of order d if (i) the rows of $\mathbf{H}$ are mutually orthogonal, and (ii) the first row and first column of $\mathbf{H}$ consist entirely of $+1$s.

Denote the i-th row of $\mathbf{H}$ by $\mathbf{h}_i^\top$. Condition (i) gives that $\langle \mathbf{h}_i, \mathbf{h}_j\rangle = 0$ for all $j \neq i$, and $\langle \mathbf{h}_i, \mathbf{h}_i\rangle = d$; together with (ii), this implies $\langle \mathbf{1}, \mathbf{h}_i\rangle = 0$ for every $i \geq 2$. It is well known (Sylvester, 1867) that a normalized Hadamard matrix of order d always exists when d is a power of 2.

Construction. Let d be the largest power of 2 at most k. The construction uses only the first d coordinates; the remaining $k-d$ coordinates carry zero weight.

First, we construct the candidate database vector using a normalized Hadamard matrix $\mathbf{H}$ of order d, and a random vector $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_d)^\top$ with $Y_1 = 1$ fixed and $Y_2, \ldots, Y_d$ independently and uniformly drawn from $\{-\delta, +\delta\}$. $\delta > 0$ is a parameter to be chosen later. Define the candidate database vector $\tilde{\mathbf{w}} \in \mathbb{R}^k$ as the following: ($\mathbf{0}$ denotes the all-zero vector of dimension $k-d$)

$$\tilde{\mathbf{w}} = \begin{pmatrix} \frac{1}{d}\mathbf{H}^\top \mathbf{Y} \\ \mathbf{0} \end{pmatrix}. \tag{2}$$

Next, we define the query sequence. For $t = 1$, we use an all-ones vector: $\mathbf{v}_1 = \mathbf{1} \in \{0,1\}^k$. Since absolute loss is non-negative, round 1's contribution to the regret is non-negative, so we obtain a valid lower bound by considering rounds $t = 2, \ldots, d$ only. For $t = 2, \ldots, d$, the queries are constructed from the rows of $\mathbf{H}$ as follows:

$$\mathbf{v}_t = \begin{pmatrix} \frac{1}{2}(\mathbf{1} + \mathbf{h}_t) \\ \mathbf{0} \end{pmatrix} \in \{0,1\}^k. \tag{3}$$

Since $\mathbf{h}_t \in \{-1,+1\}^d$, the first d coordinates of $\mathbf{v}_t$ are either 0 or 1, and the remaining $k-d$ coordinates are 0. Hence, $\mathbf{v}_t$ is a valid subset query (select a subset of the domain), the restricted class of queries within the general linear queries. The properties of this construction are captured by the three lemmas below; their proofs are deferred to Appendix C.5.

Lemma 4.2 shows that the true selectivity in each round depends only on a single random variable.

Lemma 4.2. *For each round $t \in \{2,3,\ldots,d\}$, the true selectivity of query $\mathbf{v}_t$ defined in* (3) *for $\tilde{\mathbf{w}}$ defined in* (2) *is $\sigma_t = \frac{1}{2}(1 + Y_t)$.*

Lemma 4.3 bounds the expected loss in each round.

Lemma 4.3. *For each round $t \in \{2,\ldots,d\}$, let $\hat{\sigma}_t \in [0,1]$ be the deterministic algorithm's prediction given $Y_1, \ldots, Y_{t-1}$. Let $Loss_t(\tilde{\mathbf{w}}) = |\hat{\sigma}_t - \langle \mathbf{v}_t, \tilde{\mathbf{w}}\rangle|$ be the absolute loss at round t for $\tilde{\mathbf{w}}$. Then the expected loss over Y_t for $\tilde{\mathbf{w}}$ is given by:*

$$\mathbb{E}_{Y_t}[Loss_t(\tilde{\mathbf{w}}) \mid Y_1, \ldots, Y_{t-1}] \geq \frac{\delta}{2}.$$

These lemmas show that our construction of $\tilde{\mathbf{w}}$ and $\mathbf{v}_t$ forces any deterministic algorithm to incur at least $\delta/2$ expected absolute loss in each round $t \in \{2,\ldots,d\}$. The next lemma shows that the probability of $\tilde{\mathbf{w}}$ being a valid database state depends on δ and d. This implies that by choosing δ sufficiently small, this probability can be made arbitrarily close to 1.

Lemma 4.4. *Let $E = \{\tilde{\mathbf{w}} \in \Delta_k\}$ denote the event that the candidate weight vector $\tilde{\mathbf{w}}$ defined in* (2) *lies in the simplex Δ_k. The probability that E holds is $\mathbb{P}(E) \geq 1 - 2d\exp\left(-\frac{1}{2d\delta^2}\right)$.*

We now show that using the above lemmas, we can derive the lower bound on absolute loss given in Theorem 2.1. The adversary constructs the true database vector $\mathbf{w}^* \in \Delta_k$ as follows: it draws $\mathbf{Y}$ and computes the candidate database vector $\tilde{\mathbf{w}}$ via (2). If $\tilde{\mathbf{w}} \in \Delta_k$, i.e., $\tilde{\mathbf{w}}$ is a valid database vector, it sets $\mathbf{w}^* = \tilde{\mathbf{w}}$; otherwise, it sets $\mathbf{w}^* = \frac{1}{k}\mathbf{1}$. This guarantees that $\mathbf{w}^*$ is always valid. We re-use notation from Lemma 4.4 and denote $E = \{\tilde{\mathbf{w}} \in \Delta_k\}$, and its complement by E^c.

The query sequence is given by (3). Let $\text{Loss}_t(\mathbf{w}^*) = |\hat{\sigma}_t - \langle \mathbf{v}_t, \mathbf{w}^*\rangle|$ denote the algorithm's absolute loss at round t for $\mathbf{w}^*$. Fix a deterministic algorithm and a round $t \in \{2,\ldots,d\}$. The law of total expectation gives that the expected loss for $\mathbf{w}^*$ over $\mathbf{Y}$ is:

$$\mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\mathbf{w}^*)] = \mathbb{P}(E)\cdot\mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\mathbf{w}^*) \mid E] + \mathbb{P}(E^c)\cdot\mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\mathbf{w}^*) \mid E^c].$$

The first term corresponds to the case where $\mathbf{w}^* = \tilde{\mathbf{w}}$, and the second corresponds to the case where the adversary outputs the uniform fallback ($\mathbf{w}^* = \frac{1}{k}\mathbf{1}$). Since the absolute loss is non-negative, we

can drop the second term. This yields:

$$\mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\mathbf{w}^*)] \geq \mathbb{P}(E) \cdot \mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\tilde{\mathbf{w}}) \mid E]. \tag{4}$$

To lower-bound the right-hand side, we apply the law of total expectation to the loss for the candidate database vector $\tilde{\mathbf{w}}$:

$$\mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\tilde{\mathbf{w}})] = \mathbb{P}(E) \cdot \mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\tilde{\mathbf{w}}) \mid E] + \mathbb{P}(E^c) \cdot \mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\tilde{\mathbf{w}}) \mid E^c].$$

Rearranging this equation isolates the exact term we need:

$$\mathbb{P}(E) \cdot \mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\tilde{\mathbf{w}}) \mid E] = \mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\tilde{\mathbf{w}})] - \mathbb{P}(E^c) \cdot \mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\tilde{\mathbf{w}}) \mid E^c]. \tag{5}$$

By Lemma 4.3, we have:

$$\mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\tilde{\mathbf{w}})] = \mathbb{E}_{Y_1,\ldots,Y_{t-1}}\left[\mathbb{E}_{Y_t}[\text{Loss}_t(\tilde{\mathbf{w}}) \mid Y_1, \ldots, Y_{t-1}]\right] \geq \delta/2.$$

By Lemma 4.2, $\langle \mathbf{v}_t, \tilde{\mathbf{w}} \rangle = \frac{1}{2}(1 + Y_t) \in [0, 1]$, and $\hat{\sigma}_t \in [0, 1]$, so the absolute loss satisfies $\text{Loss}_t(\tilde{\mathbf{w}}) \leq 1$ deterministically; thus $\mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\tilde{\mathbf{w}}) \mid E^c] \leq 1$. By Lemma 4.4, we know that $\mathbb{P}(E^c) \leq 2d \exp(-\frac{1}{2d\delta^2})$. Setting $\delta = \frac{1}{4\sqrt{d \ln d}}$ yields:

$$\mathbb{P}(E^c) \leq 2d \exp\left(-\frac{16d \ln d}{2d}\right) = 2d^{-7}.$$

For any integer $d \geq 2$, it is straightforward to verify that $2d^{-7} \leq \frac{1}{16\sqrt{d \ln d}} = \frac{\delta}{4}$. Substituting these bounds into (5) gives:

$$\mathbb{P}(E) \cdot \mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\tilde{\mathbf{w}}) \mid E] \geq \frac{\delta}{2} - \left(\frac{\delta}{4} \cdot 1\right) = \frac{\delta}{4}.$$

Combining with (4), we conclude that the expected loss for the true database state at every single round is at least $\delta/4$, i.e., $\mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\mathbf{w}^*)] \geq \delta/4$.

By Observation 1, the algorithm's expected regret is exactly its expected cumulative loss. Summing over the $d - 1$ query rounds indexed $t = 2, \ldots, d$ we can lower bound the regret as:

$$\mathbb{E}[\text{Regret}_T] \geq \sum_{t=2}^{d} \mathbb{E}_{\mathbf{Y}}[\text{Loss}_t(\mathbf{w}^*)] \geq (d-1) \cdot \frac{\delta}{4} = \frac{d-1}{16\sqrt{d \ln d}} = \Omega\left(\sqrt{k/\log k}\right).$$

The last step follows from the fact that d is the largest power of 2 at most k, and thus $d > k/2$.

Finally, by Yao's minimax principle, we conclude that any randomized algorithm incurs $\Omega(\sqrt{k/\log k})$ expected regret. This completes the proof of the lower bound.

5 Lower Bounds under Squared Loss for Dynamic Databases

We prove the $\Omega(\log T)$ lower bound under squared loss for dynamic databases in Theorem 2.2.

Construction. Let the support size $k = 2$; for $k > 2$, the adversary places zero mass on the remaining $k - 2$ coordinates and never queries them. The adversary fixes the query $\mathbf{v}_t = (1, 0)$ for every round t, draws a parameter $\mu \sim \text{Uniform}[0, 1]$ once before the first round, and in each round t generates the database vector $\mathbf{w}_t$ for the database as follows:

$$\mathbf{w}_t = \begin{cases} (1, 0), & \text{with prob. } \mu \\ (0, 1), & \text{with prob. } 1 - \mu. \end{cases} \tag{6}$$

Consequently, the true selectivity $\sigma_t = \langle \mathbf{v}_t, \mathbf{w}_t \rangle \in \{0, 1\}$ satisfies:

$$\sigma_t \mid \mu \overset{\text{i.i.d.}}{\sim} \text{Bernoulli}(\mu), \quad \text{and thus } \mathbb{E}[\sigma_t \mid \mu] = \mu. \tag{7}$$

Since the query under this construction is fixed, every database vector in Δ_2 makes the same prediction in all rounds, and the optimal fixed comparator is at least as good as the fixed database $(\mu, 1-\mu)$, whose loss is $\sum_t (\sigma_t - \mu)^2$. Conditional on $\mathcal{H}_t$ and μ, the prediction $\hat{\sigma}_t$ is fixed while σ_t has mean μ. Therefore, the cross term vanishes and leaves $(\hat{\sigma}_t - \mu)^2$. Hence, the expected regret is at least the sum of the expected squared error in estimating μ in each round, as shown in Lemma 5.1:

Lemma 5.1. *For every deterministic algorithm, the expected regret satisfies* $\mathbb{E}[\text{Regret}_T] \geq \sum_{t=1}^{T} \mathbb{E}\left[(\hat{\sigma}_t - \mu)^2\right]$.

Lemma 5.2 provides a lower bound on the expected squared error in estimating μ in each round by the posterior variance of μ:

Lemma 5.2. *For every round t and every deterministic algorithm,* $\mathbb{E}\big[(\hat{\sigma}_t - \mu)^2\big] \geq \mathbb{E}\big[\mathrm{Var}(\mu \mid \mathcal{H}_t)\big]$.

The posterior variance can be computed exactly. Since the uniform prior is $\mathrm{Beta}(1,1)$ and the coins are Bernoulli, the posterior is $\mathrm{Beta}(1 + H_{t-1}, t - H_{t-1})$, where H_{t-1} counts the ones among $\sigma_1, \ldots, \sigma_{t-1}$ and is uniform on $\{0, \ldots, t-1\}$; averaging the Beta variance over H_{t-1} gives Lemma 5.3. The proofs of the three lemmas are deferred to Appendix D.2.

Lemma 5.3. *For every round t,* $\mathbb{E}\big[\mathrm{Var}(\mu \mid \mathcal{H}_t)\big] = \frac{1}{6(t+1)}$.

Putting Lemmas 5.1 to 5.3 together, we get that any deterministic algorithm satisfies

$$\mathbb{E}[\mathrm{Regret}_T] \;\geq\; \sum_{t=1}^{T} \frac{1}{6(t+1)} \;\geq\; \frac{1}{6} \int_1^{T+1} \frac{dx}{x+1} \;=\; \frac{1}{6} \ln \frac{T+2}{2} \;=\; \Omega(\log T).$$

By Yao's minimax principle, the same lower bound holds for expected regret of every randomized algorithm. This completes the proof of the lower bound under squared loss for dynamic databases.

6 Experimental Results

Setup. We evaluate on *Taxi* (NYC Taxi and Limousine Commission, 2024), the NYC yellow-taxi trip-distance data of 2019–2020 (108.2M tuples), which suits both settings: the static database is the entire dataset, and the dynamic database changes in every round, consisting of the trips of one pickup date. Results for additional datasets *Power* (Dua et al., 2017) and *Forest* (Dua et al., 2017) are provided in Appendix F. We consider two changing query workloads: under the *oscillating* workload, the query region moves smoothly back and forth across the domain, causing the workload to evolve continuously; under the *abrupt* workload, queries concentrate on a fixed region within a phase and jump to a new region between phases, yielding sharp transitions. We draw five independent query streams for each setting and report the average. Detailed settings, implementation details, and additional results are provided in Appendix F.

Algorithms and baselines. Following the upper regret bounds we obtained in different settings, we evaluate the SeqMaxEnt algorithm (Algorithm 1) on static databases; on dynamic databases, we evaluate the Online Newton Step (ONS) algorithm (Hazan et al., 2007) under squared loss, a computationally tractable alternative to EWOO that also achieves logarithmic regret in T (Corollary D.5 in the appendix), and the FTRL algorithm with negative-entropy regularization under absolute loss. The details of the implementation are given in the appendix. We compare with three baselines that operate using query feedback only: NeuroCDF (Wu et al., 2025), MSCN (Kipf et al., 2019), and PtsHist (Hu et al., 2022). Since none of the baselines are designed for continual adaptation, we periodically refit each baseline from scratch on all observed query–selectivity pairs every 20 rounds; results for frozen baselines are in the appendix. The comparison in Table 1 uses range queries, which all baseline interfaces support. We report regret results of general linear queries in Figure 1 and defer those of range queries to the appendix.

Comparison with baselines. Table 1 compares all methods on identical range-query streams. We report cumulative squared and absolute loss to compare estimation accuracy, and the 90th-percentile *Q-error* (Moerkotte et al., 2009) to assess relative estimation accuracy. It is defined as the 90th percentile of $\max\{\hat{\sigma}_t(R_i), \sigma_t(R_i)\} / \min\{\hat{\sigma}_t(R_i), \sigma_t(R_i)\}$ over the evaluated queries, where $\hat{\sigma}_t(R_i)$ and $\sigma_t(R_i)$ are the predicted and true selectivities of query R_i, respectively. On the static database, SeqMaxEnt achieves the lowest error on every metric: its cumulative squared loss is about $18\times$ and $26\times$ lower than that of the strongest baseline, NeuroCDF, under the abrupt and oscillating workloads, respectively, and its Q-errors of 1.006 and 1.003 indicate accurate relative predictions. On the dynamic database, ONS achieves the lowest error on all three metrics under the abrupt workload, with $5.7\times$ lower squared loss than NeuroCDF; under the oscillating workload, however, NeuroCDF attains lower squared and absolute loss and PtsHist a lower Q-error. Periodic refitting thus remains competitive when queries repeatedly target overlapping regions, whereas per-query updates are particularly beneficial under abrupt workload changes.

Regret as k and T vary (Figure 1). As the support size grows from $k = 20$ to 200, the squared-loss regret of SeqMaxEnt on the static database (Figures 1a and 1b) rises only from 0.006 to 0.096

Table 1: Selectivity estimation accuracy on the static and dynamic Taxi databases under abrupt and oscillating workloads (range queries, $k = 200$, $T = 500$): cumulative squared and absolute loss and 90th-percentile Q-error; the best value in each column is in bold.

	static, abrupt			static, oscillating			dynamic, abrupt			dynamic, oscillating		
Method	sq. loss	abs. loss	q-err$_{90}$	sq. loss	abs. loss	q-err$_{90}$	sq. loss	abs. loss	q-err$_{90}$	sq. loss	abs. loss	q-err$_{90}$
SeqMaxEnt	**0.133**	**0.893**	**1.006**	**0.0009**	**0.099**	**1.003**	–	–	–	–	–	–
ONS	0.321	4.71	1.341	0.120	3.95	1.246	**0.455**	**7.55**	**1.653**	0.225	6.36	1.550
FTRL	1.43	10.4	2.288	0.338	5.59	1.350	1.44	11.8	2.236	0.375	7.22	1.527
NeuroCDF	2.45	7.59	8.128	0.023	2.41	1.593	2.61	10.2	3.918	**0.144**	**5.39**	1.784
MSCN	2.77	10.6	68.6	0.076	3.60	1.794	2.84	12.5	359	0.241	6.53	3.290
PtsHist	5.29	10.8	1.183	0.119	1.83	1.093	5.33	14.4	1.829	0.263	5.84	**1.390**

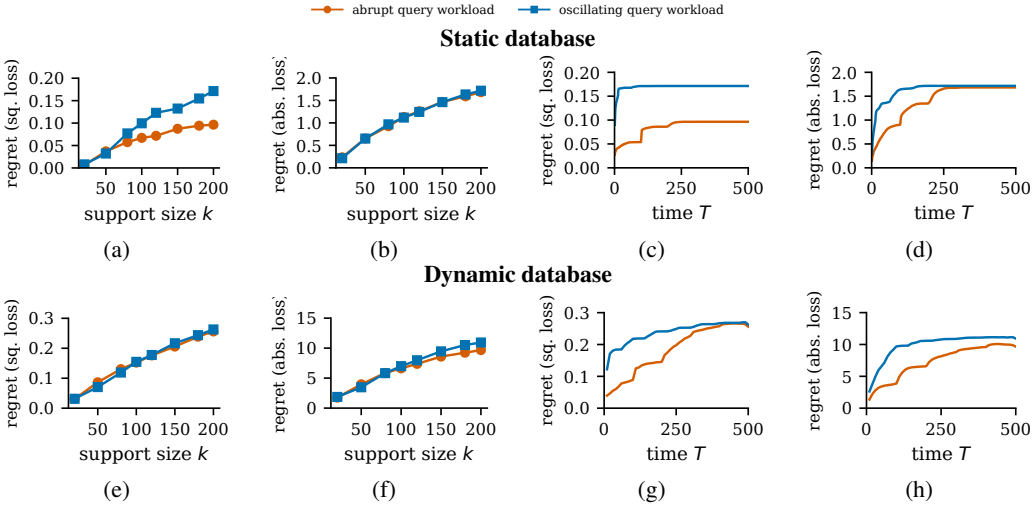

Figure 1: Regret on the static (top; SeqMaxEnt) and dynamic (bottom; ONS under squared loss, FTRL under absolute loss) Taxi databases, as k varies with $T = 500$ (a, b, e, f) and along 500-round streams with $k = 200$ (c, d, g, h).

under the abrupt workload and from 0.008 to 0.171 under the oscillating workload, far below the worst-case bound $2 \ln k$ of Section 3, and its absolute-loss regret behaves similarly; on the dynamic database, the regret of ONS under squared loss rises from about 0.03 to 0.26 and that of FTRL under absolute loss from about 1.8 to 9.7–10.9 (Figures 1e and 1f). Along the query stream, the regret on the static database plateaus once enough feedback has accumulated, consistent with our regret bounds, which are independent of T (Figures 1c and 1d). On the dynamic database, the oscillating workload incurs most of its regret early, whereas the abrupt workload accumulates regret in steps, each following a jump of the queried region, and in both cases the growth slows down late in the stream, with the regret even decreasing slightly although the database keeps changing (Figures 1g and 1h). This is an effect of the comparator: regret is measured against the best *fixed* histogram in hindsight, and the pandemic-era demand collapse from round 441 onward (mid-March 2020) forces this histogram to fit two regimes at once, so its loss grows faster than that of the learners, which simply track the new regime.

7 Conclusions and Future Work

In this paper, we modeled the selectivity estimation problem for linear queries as an online learning problem, in which datasets and query workloads may change arbitrarily over time without distributional assumptions, and the learner observes only query–selectivity feedback. We established (nearly) tight regret bounds for both static and dynamic databases, under both squared and absolute loss functions. Several interesting directions remain open for future work: (i) closing the gap between the $\Omega(\log T)$ lower bound and $O(k \log T)$ upper bound for dynamic databases under squared loss; (ii) extending to richer query classes such as join queries or queries with multiple predicates; and (iii) studying dynamic regret against a slowly changing database.

AI USE STATEMENT

Generative AI tools were used to improve the clarity and readability of the manuscript, assist with literature searches (all references are manually verified), and provide coding assistance for portions of the experimental implementation (code is manually verified). We did not use generative AI tools to develop theoretical models or conceptual frameworks, formulate mathematical claims, develop or write proofs, propose or refine research hypotheses, synthesize datasets, or interpret the experimental results.

The authors reviewed all AI-assisted work and take full responsibility for the final content of this work, including the manuscript, code, and reported results.

REFERENCES

Aaron Anderson and Michael Benedikt. From learnable objects to learnable random objects. *arXiv preprint arXiv:2504.00847*, 2025.

Daniel Berend and Aryeh Kontorovich. A sharp estimate of the binomial mean absolute deviation with applications. *Statistics & Probability Letters*, 83(4):1254–1259, 2013.

James O Berger. *Statistical decision theory and Bayesian analysis*. Springer Science & Business Media, 2013.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Sébastien Bubeck, Bo'az Klartag, Yin Tat Lee, Yuanzhi Li, and Mark Sellke. Chasing nested convex bodies nearly optimally. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1496–1508. SIAM, 2020.

Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

Yuvaraj Chesetti and Prashant Pandey. To adapt or not to adapt, that is the ski question. *Proceedings of the ACM on Management of Data*, 4(3 (SIGMOD):1–27, 2026.

Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

Imre Csiszár and János Körner. *Information theory: coding theorems for discrete memoryless systems*. Cambridge University Press, 2011.

Dheeru Dua, Casey Graff, et al. Uci machine learning repository. 2017.

John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *Proceedings of the 25th International Conference on Machine Learning*, pp. 272–279, 2008.

Anshuman Dutt, Chi Wang, Azade Nazi, Srikanth Kandula, Vivek R. Narasayya, and Surajit Chaudhuri. Selectivity estimation for range predicates using lightweight models. In *Proceedings of the VLDB Endowment*, volume 12, pp. 1044–1057, 2019.

Joel Friedman and Nathan Linial. On convex body chasing. *Discret. Comput. Geom.*, 9:293–321, 1993. doi: 10.1007/BF02189324. URL https://doi.org/10.1007/BF02189324.

Elad Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3-4):157–325, 2016.

Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2–3):169–192, 2007.

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

Kathy Horadam. *Hadamard matrices and their applications*. Princeton university press, 2012.

Xiao Hu, Yuxi Liu, Haibo Xiu, Pankaj K Agarwal, Debmalya Panigrahi, Sudeepa Roy, and Jun Yang. Selectivity functions of range queries are learnable. In *Proceedings of the 2022 International Conference on Management of Data*, pp. 959–972, 2022.

Alicia A Johnson, Miles Q Ott, and Mine Dogucu. *Bayes rules!: An introduction to applied Bayesian modeling*. Chapman and Hall/CRC, 2022.

Kyoungmin Kim, Jisung Jung, In Seo, Wook-Shin Han, Kangwoo Choi, and Jaehyok Chong. Learned cardinality estimation: An in-depth study. In *Proceedings of the 2022 international conference on management of data*, pp. 1214–1227, 2022.

Andreas Kipf, Thomas Kipf, Bernhard Radke, Viktor Leis, Peter A. Boncz, and Alfons Kemper. Learned cardinalities: Estimating correlated joins with deep learning. In *9th Biennial Conference on Innovative Data Systems Research (CIDR)*, 2019.

Jyrki Kivinen and Manfred K Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and computation*, 132(1):1–63, 1997.

Dieter Kraft. A software package for sequential quadratic programming. *Forschungsbericht-Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt*, 1988.

Tim Kraska, Alex Beutel, Ed H. Chi, Jeffrey Dean, and Neoklis Polyzotis. The case for learned index structures. In *Proceedings of the 2018 ACM SIGMOD International Conference on Management of Data*, pp. 489–504. ACM, 2018.

Ani Kristo, Kapil Vaidya, Ugur Çetintemel, Sanchit Misra, and Tim Kraska. The case for a learned sorting algorithm. In *Proceedings of the 2020 ACM SIGMOD international conference on management of data*, pp. 1001–1016, 2020.

Beibin Li, Yao Lu, and Srikanth Kandula. Warper: Efficiently adapting learned cardinality estimators to data and workload drifts. In *Proceedings of the 2022 International Conference on Management of Data*, pp. 1920–1933, 2022.

Pengfei Li, Wenqing Wei, Rong Zhu, Bolin Ding, Jingren Zhou, and Hua Lu. Alece: An attention-based learned cardinality estimator for spj queries on dynamic workloads. *Proceedings of the VLDB Endowment*, 17(2):197–210, 2023.

Ryan Marcus, Parimarjan Negi, Hongzi Mao, Chi Zhang, Mohammad Alizadeh, Tim Kraska, Olga Papaemmanouil, and Nesime Tatbul. Neo: A learned query optimizer. *arXiv preprint arXiv:1904.03711*, 2019.

Guido Moerkotte, Thomas Neumann, and Gabriele Steidl. Preventing bad plans by bounding the impact of cardinality estimation errors. *Proceedings of the VLDB Endowment*, 2(1):982–993, 2009. doi: 10.14778/1687627.1687738.

Parimarjan Negi, Ziniu Wu, Andreas Kipf, Nesime Tatbul, Ryan Marcus, Sam Madden, Tim Kraska, and Mohammad Alizadeh. Robust query driven cardinality estimation under changing workloads. *Proceedings of the VLDB Endowment*, 16(6):1520–1533, 2023.

NYC Taxi and Limousine Commission. Tlc trip record data, 2024. URL `https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page`.

Yongjoo Park, Shucheng Zhong, and Barzan Mozafari. Quicksel: Quick selectivity learning with mixture models. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pp. 1017–1033, 2020.

Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

Elias M. Stein and Rami Shakarchi. *Real Analysis: Measure Theory, Integration, and Hilbert Spaces*, volume 3 of *Princeton Lectures in Analysis*. Princeton University Press, Princeton, NJ, 2005. ISBN 978-0-691-11387-6.

Ji Sun, Jintao Zhang, Zhaoyan Sun, Guoliang Li, and Nan Tang. Learned cardinality estimation: A design space exploration and a comparative evaluation. *Proceedings of the VLDB Endowment*, 15 (1):85–97, 2021.

James Joseph Sylvester. Lx. thoughts on inverse orthogonal matrices, simultaneous signsuccessions, and tessellated pavements in two or more colours, with applications to newton's rule, ornamental tile-work, and the theory of numbers. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 34(232):461–475, 1867.

Xiaoying Wang, Changbo Qu, Weiyuan Wu, Jiannan Wang, and Qingqing Zhou. Are we ready for learned cardinality estimation? In *Proceedings of the VLDB Endowment*, volume 14, pp. 1640–1654, 2021.

Zilong Wang, Qixiong Zeng, Ning Wang, Haowen Lu, and Yue Zhang. Ceda: learned cardinality estimation with domain adaptation. *Proceedings of the VLDB Endowment*, 16(12):3934–3937, 2023.

Peizhi Wu and Zachary G Ives. Modeling shifting workloads for learned database systems. *Proceedings of the ACM on Management of Data*, 2(1):1–27, 2024.

Peizhi Wu, Haoshu Xu, Ryan Marcus, and Zachary G. Ives. A practical theory of generalization in selectivity learning. *Proc. VLDB Endow.*, 18(6):1811–1824, February 2025. ISSN 2150-8097. doi: 10.14778/3725688.3725708. URL https://doi.org/10.14778/3725688.3725708.

Zongheng Yang, Amog Kamsetty, Sifei Luan, Eric Liang, Yan Duan, Xi Chen, and Ion Stoica. NeuroCard: One cardinality estimator for all tables. In *Proceedings of the VLDB Endowment*, volume 14, pp. 61–73, 2020.

Andrew Chi-Chin Yao. Probabilistic computations: Toward a unified measure of complexity. In *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*, pp. 222–227. IEEE, 1977.

Sepanta Zeighami and Cyrus Shahabi. Theoretical analysis of learned database operations under distribution shift through distribution learnability. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024a.

Sepanta Zeighami and Cyrus Shahabi. Towards establishing guaranteed error for learned database operations. In *The Twelfth International Conference on Learning Representations*, 2024b.

Tianjing Zeng, Junwei Lan, Jiahong Ma, Wenqing Wei, Rong Zhu, Yingli Zhou, Pengfei Li, Bolin Ding, Defu Lian, Zhewei Wei, et al. Price: A pretrained model for cross-database cardinality estimation. *Proceedings of the VLDB Endowment*, 18(3):637–650, 2024.

Jiaqi Zhu, Shaofeng Cai, Yanyan Shen, Gang Chen, Fang Deng, and Beng Chin Ooi. In-context adaptation to concept drift for learned database operations. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (icml-03)*, pp. 928–936, 2003.

A Related Work

Online Learning and Online Convex Optimization. Online learning is a well-established framework for sequential decision-making under uncertainty (e.g., (Cesa-Bianchi & Lugosi, 2006; Shalev-Shwartz, 2012; Hazan, 2016; Hazan et al., 2007; Zinkevich, 2003; Kivinen & Warmuth, 1997)). Our selectivity estimation problem fits the framework of online convex optimization (OCO): each query induces a convex loss over the probability simplex, even though the underlying database state is never observed by the learner. Unlike generic OCO analyses, our results exploit the specific query structure and simplex geometry to derive problem-dependent upper and lower bounds.

Online Convex Body Chasing. Our static database setting is also related to the convex body chasing problem from online algorithms (Friedman & Linial, 1993). In nested convex body chasing, an online algorithm observes nested convex sets, chooses a point from each newly revealed set, and pays movement cost equal to the distance between consecutive chosen points. Our static database setting has a similar structure: each observed query-selectivity pair shrinks the set of candidate database vectors consistent with the history, from the entire k-dimensional simplex to the true database. However, the performance measure is different: we measure scalar prediction error based on projections along the query vector, rather than movement cost, so the algorithms and lower bounds from convex body chasing do not directly apply in our setting. Nevertheless, our use of Hadamard matrices in the lower bound construction was inspired by similar techniques for online convex body chasing (Bubeck et al., 2020).

Learning-Based Selectivity Estimation. Machine learning for selectivity (cardinality) estimation has been extensively studied (Dutt et al., 2019; Yang et al., 2020; Li et al., 2023; Park et al., 2020; Wang et al., 2023; Sun et al., 2021; Kim et al., 2022; Wang et al., 2021), including systems for handling data and query workload drift (Li et al., 2022; Negi et al., 2023; Wu & Ives, 2024; Zeng et al., 2024). On the theoretical side, Hu et al. (2022) established PAC sample-complexity bounds for range queries under i.i.d. assumptions, and (Zeighami & Shahabi, 2024b;a; Wu et al., 2025) develop generalization guarantees for learned database operations. Zhu et al. (2025) use in-context learning to adapt learned database operations to concept drift. Their cardinality estimator combines query feedback with histograms of database attributes, and their generalization bounds assume database records drawn i.i.d. from a continuous distribution. These results are distribution-dependent, whereas our work provides an *online* framework using only query feedback, where data and queries may evolve adversarially, with regret bounds that hold uniformly without distributional assumptions.

Online Learning in Databases. Recent work (Chesetti & Pandey, 2026) applies online-algorithmic ideas to design adaptive filters, addressing a different database problem from selectivity estimation. From a more abstract learning-theoretic perspective, Anderson & Benedikt (2025) study how learnability transfers from base function classes to derived statistical function classes in both PAC and online learning settings, which is motivated partly by query processing (Hu et al., 2022). Their agnostic and realizable online cases relate to our dynamic and static database settings, respectively; in particular, their preservation results yield a generic, non-constructive regret upper bound for the agnostic case. Our contribution is problem-specific: for both static and dynamic databases, we give upper and matching lower bounds on regret with concrete algorithms across different query classes and loss functions.

B Query Subclasses and Summary of Results

This section defines the point, range, and subset query classes introduced in Section 2.1 and summarizes the regret bounds for these classes and the full linear query class $\mathcal{Q}_{\text{linear}} := [0,1]^k$. The results extend the summary in Section 2.2 with sharper bounds for point queries.

Query subclasses. Recall that a linear query is a vector $\mathbf{v} = (v_1, \ldots, v_k) \in [0,1]^k$ whose selectivity on a database $\mathbf{w} \in \Delta_k$ is $\sigma(\mathbf{v}, \mathbf{w}) = \langle \mathbf{v}, \mathbf{w} \rangle$. We note that linear queries include commonly studied query classes:

- *Subset queries* $\mathcal{Q}_{\text{subset}} := \{0,1\}^k$, whose selectivity captures the cumulative relative frequency of a set of elements.

Table 2: Regret bounds for selectivity estimation on **static** databases. k is the support size of the domain. Each row reports upper and lower regret bounds holding for the indicated query classes. The shaded cell is proved in Sections 3 and 4; all other cells are proved in Section C.

Loss	Query Class	Lower Bound	Upper Bound	Algorithm
Squared	Point	$1/4$	1	CoordMem (Algorithm 2)
	Range, Subset, Linear	$\Omega(\log k)$	$O(\log k)$	SeqMaxEnt (Algorithm 1)
Absolute	Point	$1/2$	1	CoordMem (Algorithm 2)
	Subset, Linear	$\Omega\left(\sqrt{k/\log k}\right)$	$O\left(\sqrt{k \log k}\right)$	SeqMaxEnt (Algorithm 1)

Table 3: Regret bounds for selectivity estimation on **dynamic** databases. T is the number of rounds and k is the support size of the domain. Each row reports upper and lower regret bounds holding for the indicated query classes. The shaded cell is proved in Section 5; all other cells are proved in Sections D and E.

Loss	Query Class	Lower Bound	Upper Bound	Algorithm
Squared	Point, Range, Subset, Linear	$\Omega(\log T)$	$O(k \log T)$	EWOO (Algorithm 3)
Absolute	Point	$\Omega(\sqrt{T})$	$O(\sqrt{T})$	FTRL (ℓ_2 regularizer) (Algorithm 4)
	Subset, Linear	$\Omega\left(\sqrt{T \log k}\right)$	$O\left(\sqrt{T \log k}\right)$	FTRL (neg-entropy regularizer) (Algorithm 4)

- *Range queries* $\mathcal{Q}_{\text{range}} := \left\{\mathbf{v} \in \{0,1\}^k \;\middle|\; v_i = 1 \text{ if } a \le i \le b \text{ for some } a \le b\right\}$, which is a special case of subset queries where the set of indices corresponds to a range of (ordered) elements.
- *Point queries* $\mathcal{Q}_{\text{point}} := \left\{\mathbf{v} \in \{0,1\}^k \;\middle|\; v_i = 1 \text{ if } i = a \text{ for some } a\right\}$, which further refines range queries and isolates the relative frequency of a specific element.

These query classes are nested as $\mathcal{Q}_{\text{point}} \subset \mathcal{Q}_{\text{range}} \subset \mathcal{Q}_{\text{subset}} \subset \mathcal{Q}_{\text{linear}}$.

Transfer of bounds between nested classes. The nesting has two consequences that we use throughout. An upper bound for a query class also holds for each of its subclasses. A lower bound for a subclass also holds for every class containing it, since the same adversarial construction remains valid. Thus, our upper bounds for linear queries apply to all three subclasses, and our lower bounds for point, range, or subset queries also apply to the larger classes. We additionally obtain sharper upper bounds specifically for point queries.

Summary of results. Tables 2 and 3 summarize our regret bounds by database setting, loss function, and query class, including the bounds for subclasses that are not stated in the main body. For static databases under squared loss, the optimal regret is $\Theta(\log k)$ for range, subset, and linear queries: the upper bound is analyzed in Section 3, with deferred proofs in Section C.2, and the lower bound is proved in Section C.4. Under absolute loss, the optimal regret is $\tilde{\Theta}(\sqrt{k})$ for subset and linear queries, with the upper bound proved in Section C.3 and the lower bound established in Section 4, whose supporting lemmas are proved in Section C.5. Point queries admit constant regret under both loss functions (Section C.6). For dynamic databases under squared loss, the $O(k \log T)$ upper bound for all four query classes is proved in Section D.1, and the $\Omega(\log T)$ lower bound is established in Section 5, with deferred proofs in Section D.2. Under absolute loss, point queries admit $\Theta(\sqrt{T})$ regret, with lower and upper bounds proved in Sections E.1.1 and E.2.1, respectively; subset and linear queries admit $\Theta(\sqrt{T \log k})$ regret, with lower and upper bounds proved in Sections E.1.2 and E.2.2, respectively.

C Missing Proofs for Static Databases

This section gives the deferred proofs and additional query-subclass results for static databases.

C.1 Proof of Observation 1 in Section 2

Observation 1. *For a static database, the regret of any algorithm $\mathcal{A}$ equals its cumulative loss: $\text{Regret}_T(\mathcal{A}) = \sum_{t=1}^{T} \text{Loss}_t$. Moreover, there exists an online algorithm whose regret is at most $k-1$ under either squared or absolute loss, uniformly over all horizons T. Thus, the regret upper bound is independent of T.*

Proof. The fixed database $\mathbf{w}^*$ incurs zero comparator loss, so (1) gives $\text{Regret}_T(\mathcal{A}) = \sum_{t=1}^{T} \text{Loss}_t$ for every algorithm $\mathcal{A}$. For the upper bound, choose in each round a database estimate $\hat{\mathbf{w}}_t \in \Delta_k$ consistent with all previous observations, and predict $\langle \mathbf{v}_t, \hat{\mathbf{w}}_t \rangle$; such an estimate exists because $\mathbf{w}^*$ satisfies these constraints. If $\mathbf{v}_t$ lies in the span of $\mathbf{1}, \mathbf{v}_1, \ldots, \mathbf{v}_{t-1}$, the prediction is exact because $\hat{\mathbf{w}}_t$ and $\mathbf{w}^*$ agree on the normalization and all previous query constraints. Each round with nonzero loss therefore increases the rank of these constraints. The rank starts at 1 and is at most k, so there are at most $k-1$ such rounds, each with loss at most 1 under either loss function. Hence the regret is at most $k-1$ for every horizon T. □

C.2 Deferred Proofs for Section 3: Upper Bound under Squared Loss

We first state the standard inequalities used in the analysis:

Lemma C.1 (Pinsker's Inequality (Csiszár & Körner, 2011))**.** *For any two probability distributions $\mathbf{p}, \mathbf{q}$ over a finite domain, the L_1 norm of the difference satisfies:*

$$\|\mathbf{p} - \mathbf{q}\|_1 \leq \sqrt{2 D_{\mathrm{KL}}(\mathbf{p} \| \mathbf{q})}, \tag{8}$$

where $D_{\mathrm{KL}}(\mathbf{p}\|\mathbf{q}) = \sum_{i=1}^{k} p_i \ln(\frac{p_i}{q_i})$ is the KL-divergence between $\mathbf{p}$ and $\mathbf{q}$. A summand is defined as zero when $p_i = 0$, and as $+\infty$ when $p_i > 0$ and $q_i = 0$.

Lemma C.2 (Pythagorean Inequality for KL Divergence (Cover, 1999))**.** *Let $\mathcal{C} \subseteq \Delta_k$ be nonempty, closed, and convex, and let $\mathbf{q} \in \Delta_k$ satisfy $q_i > 0$ for every $i \in [k]$. The minimizer*

$$\mathbf{p}^* = \arg\min_{\mathbf{p} \in \mathcal{C}} D_{\mathrm{KL}}(\mathbf{p}\|\mathbf{q})$$

exists and is unique. For every $\mathbf{p} \in \mathcal{C}$,

$$D_{\mathrm{KL}}(\mathbf{p}\|\mathbf{q}) \geq D_{\mathrm{KL}}(\mathbf{p}\|\mathbf{p}^*) + D_{\mathrm{KL}}(\mathbf{p}^*\|\mathbf{q}). \tag{9}$$

Lemma C.3 (Hölder's Inequality (Stein & Shakarchi, 2005))**.** *Let $(\Omega, \mathcal{F}, \mu)$ be a measure space, and let $1 < p, q < \infty$ satisfy*

$$\frac{1}{p} + \frac{1}{q} = 1.$$

If f and g are measurable functions on Ω, then

$$\int_\Omega |fg|\, d\mu \leq \left(\int_\Omega |f|^p\, d\mu \right)^{1/p} \left(\int_\Omega |g|^q\, d\mu \right)^{1/q}.$$

Moreover, if f is bounded, $f \in L^\infty(\Omega)$ and $g \in L^1(\Omega)$, then

$$\int_\Omega |fg|\, d\mu \leq \|f\|_{L^\infty(\Omega)}\, \|g\|_{L^1(\Omega)}.$$

Then we prove Lemmas 3.2 and 3.3 and Proposition 3.1:

Lemma 3.2. *For any round $t \in \{1, 2, \ldots, T\}$, the squared loss incurred by the* SeqMaxEnt *algorithm satisfies $\text{Loss}_t \leq 2\, D_{KL}(\hat{\mathbf{w}}_{t+1} \| \hat{\mathbf{w}}_t)$.*

Proof. The squared loss in round t is $\text{Loss}_t = (\sigma_t - \hat{\sigma}_t)^2 = (\sigma_t - \langle \mathbf{v}_t, \hat{\mathbf{w}}_t \rangle)^2$. Since $\hat{\mathbf{w}}_{t+1} \in \mathcal{C}_{t+1}$, it satisfies the most recent equality $\langle \mathbf{v}_t, \hat{\mathbf{w}}_{t+1} \rangle = \sigma_t$. Substituting this into the squared loss,

$$\text{Loss}_t = \big(\langle \mathbf{v}_t, \hat{\mathbf{w}}_{t+1} \rangle - \langle \mathbf{v}_t, \hat{\mathbf{w}}_t \rangle \big)^2 = \langle \mathbf{v}_t, \hat{\mathbf{w}}_{t+1} - \hat{\mathbf{w}}_t \rangle^2.$$

By Lemma C.3 and $\|\mathbf{v}_t\|_\infty \le 1$ (as $\mathbf{v}_t \in [0,1]^k$), and then by Lemma C.1,

$$\text{Loss}_t \;\le\; \|\mathbf{v}_t\|_\infty^2 \,\|\hat{\mathbf{w}}_{t+1} - \hat{\mathbf{w}}_t\|_1^2 \;\le\; \|\hat{\mathbf{w}}_{t+1} - \hat{\mathbf{w}}_t\|_1^2 \;\le\; 2\, D_{\mathrm{KL}}(\hat{\mathbf{w}}_{t+1}\|\hat{\mathbf{w}}_t). \qquad \square$$

Lemma 3.3. *For every round t, $D_{KL}(\hat{\mathbf{w}}_{t+1}\|\hat{\mathbf{w}}_t) \;\le\; D_{KL}(\hat{\mathbf{w}}_{t+1}\|\hat{\mathbf{w}}_1) \;-\; D_{KL}(\hat{\mathbf{w}}_t\|\hat{\mathbf{w}}_1)$.*

Proof. $\mathbf{u} := (1/k, \ldots, 1/k) = \hat{\mathbf{w}}_1$ denotes the uniform vector. Since $D_{\mathrm{KL}}(\mathbf{w}\|\mathbf{u}) = \ln k + \sum_i w_i \ln w_i$ for every $\mathbf{w} \in \Delta_k$, maximizing the entropy over $\mathcal{C}_t$ is the same as minimizing $D_{\mathrm{KL}}(\cdot\|\mathbf{u})$ over $\mathcal{C}_t$: the SEQMAXENT estimate is the KL projection of the uniform vector onto the current feasible region,

$$\hat{\mathbf{w}}_t = \arg\min_{\mathbf{w}\in\mathcal{C}_t} D_{\mathrm{KL}}(\mathbf{w}\|\mathbf{u}) \qquad \text{for every } t \ge 1. \tag{10}$$

Each $\mathcal{C}_t$ is nonempty (it contains $\mathbf{w}^*$), closed and convex (it is the intersection of Δ_k with finitely many hyperplanes). The uniform vector $\mathbf{u}$ has full support. Thus, $\hat{\mathbf{w}}_t$ is the KL projection of $\mathbf{u}$ onto $\mathcal{C}_t$. Since $\mathcal{C}_{t+1} \subseteq \mathcal{C}_t$, the next estimate $\hat{\mathbf{w}}_{t+1}$ also lies in $\mathcal{C}_t$. Applying Lemma C.2 with $\mathcal{C} = \mathcal{C}_t$, $\mathbf{q} = \mathbf{u}$, $\mathbf{p}^* = \hat{\mathbf{w}}_t$ and $\mathbf{p} = \hat{\mathbf{w}}_{t+1}$ gives

$$D_{\mathrm{KL}}(\hat{\mathbf{w}}_{t+1}\|\mathbf{u}) \;\ge\; D_{\mathrm{KL}}(\hat{\mathbf{w}}_{t+1}\|\hat{\mathbf{w}}_t) + D_{\mathrm{KL}}(\hat{\mathbf{w}}_t\|\mathbf{u}),$$

which is the claim since $\mathbf{u} = \hat{\mathbf{w}}_1$. $\square$

Proposition 3.1. *Under squared loss in the static database setting, the* SEQMAXENT *algorithm has regret at most* $2\ln k$.

Proof. By Lemma 3.3, summing over $t = 1, \ldots, T$ gives

$$\sum_{t=1}^{T} D_{\mathrm{KL}}(\hat{\mathbf{w}}_{t+1}\|\hat{\mathbf{w}}_t) \le D_{\mathrm{KL}}(\hat{\mathbf{w}}_{T+1}\|\hat{\mathbf{w}}_1) - D_{\mathrm{KL}}(\hat{\mathbf{w}}_1\|\hat{\mathbf{w}}_1) \le \ln k,$$

where the last inequality uses that $\hat{\mathbf{w}}_1$ is uniform, so $D_{\mathrm{KL}}(\mathbf{w}\|\hat{\mathbf{w}}_1) \le \ln k$ for every $\mathbf{w} \in \Delta_k$, and $D_{\mathrm{KL}}(\hat{\mathbf{w}}_1\|\hat{\mathbf{w}}_1) = 0$. Combining this with Lemma 3.2,

$$\text{Regret}_T = \sum_{t=1}^{T} \text{Loss}_t \le 2\sum_{t=1}^{T} D_{\mathrm{KL}}(\hat{\mathbf{w}}_{t+1}\|\hat{\mathbf{w}}_t) \le 2\ln k. \qquad \square$$

C.3 DEFERRED PROOFS FOR SECTION 3: UPPER BOUND UNDER ABSOLUTE LOSS

Corollary C.4. *The* SEQMAXENT *algorithm (Algorithm 1) achieves Regret$_T \le \sqrt{2(k-1)\ln k} = O(\sqrt{k\log k})$ for static databases under absolute loss, with general linear queries, for every horizon T.*

Proof. The SEQMAXENT algorithm incurs nonzero loss in at most $k-1$ rounds by Observation 1, and its cumulative squared loss is at most $2\ln k$ by Proposition 3.1. Applying the Cauchy–Schwarz inequality to the rounds with nonzero loss gives

$$\text{Regret}_T = \sum_{t=1}^{T} |\sigma_t - \hat{\sigma}_t| \le \sqrt{(k-1)\sum_{t=1}^{T}(\sigma_t - \hat{\sigma}_t)^2} \le \sqrt{2(k-1)\ln k}. \qquad \square$$

Corollary C.4 gives the upper bound under absolute loss in Theorem 2.1.

C.4 LOWER BOUND UNDER SQUARED LOSS IN THEOREM 2.1

We prove the $\Omega(\log k)$ regret lower bound under squared loss for static databases in Theorem 2.1. The bound holds even for range queries, and therefore also for subset and linear queries.

Proof of the squared-loss lower bound in Theorem 2.1. By Yao's minimax principle Yao (1977), to lower-bound the worst-case expected regret of any randomized algorithm, it suffices to construct an input distribution on which every deterministic algorithm incurs expected regret $\Omega(\log k)$.

Adversary construction. Let $r = \lfloor \log_2 k \rfloor$ and $k^* = 2^r$, i.e., k^* is the largest power of 2 less than or equal to k. Fix any time horizon $T \geq r$. The adversary uses only the first k^* coordinates, and the remaining $k - k^*$ coordinates carry zero mass and are never queried. Before the game starts, the adversary samples an index i^* from $[k^*]$ uniformly at random, and sets the static database vector to $\mathbf{w}^* = \mathbf{e}_{i^*}$. Thus all mass is on a single unknown index.

For each fixed value of i^*, the query sequence is deterministic and constructed via a sequence of candidate sets as follows. For each round $t \in [r]$, let $S_t \subseteq [k^*]$ be the candidate set of indices that could still be the true index i^* given observations before t. Initially, $S_1 = [k^*]$. Let $S'_t \subset S_t$ be the first half of S_t, i.e., the subset containing the $\frac{|S_t|}{2}$ smallest indices. Then the next candidate set is:

$$S_{t+1} = \begin{cases} S'_t, & i^* \in S'_t, \\ S_t \setminus S'_t, & \text{otherwise.} \end{cases}$$

A simple induction shows that each S_t is a contiguous interval of size 2^{r-t+1}, and $i^* \mid S_t \sim \mathrm{Uniform}(S_t)$: the base case $t = 1$ holds by construction; for the inductive step, assume the claim holds at round t. Then S_{t+1} is one of the two equal-sized halves of S_t that contains i^*, and conditioning a uniform distribution on a subset preserves uniformity.

Then the adversary issues the range query $\mathbf{v}_t$ such that $v_{t,i} = 1$ if $i \in S'_t$ and 0 otherwise. The true selectivity is:

$$\sigma_t = \begin{cases} 1, & i^* \in S'_t, \\ 0, & \text{otherwise.} \end{cases}$$

Since $i^* \mid \{i^* \in S_t\} \sim \mathrm{Uniform}(S_t)$ and S'_t contains exactly half of the elements of S_t, we conclude $\sigma_t \mid S_t \sim \mathrm{Bernoulli}(1/2)$.

Per-round expected loss. For any deterministic algorithm $\mathcal{A}$ and a round $t \in [r]$, its prediction $\hat{\sigma}_t$ depends only on the history $\mathcal{H}_t := (\mathbf{v}_1, \sigma_1, \ldots, \mathbf{v}_{t-1}, \sigma_{t-1}, \mathbf{v}_t)$. For any $s < t$, each σ_s encodes which half of S_s contains i^*; thus the past selectivities $\sigma_1, \ldots, \sigma_{t-1}$ uniquely determine S_t. Combined with $i^* \mid S_t \sim \mathrm{Uniform}(S_t)$ and $|S'_t| = |S_t|/2$, we have $\sigma_t \mid \mathcal{H}_t \sim \mathrm{Bernoulli}(1/2)$.

Therefore, the expected loss in each round t over all possible choices of i^* is:

$$\mathbb{E}\big[(\sigma_t - \hat{\sigma}_t)^2 \mid \mathcal{H}_t\big] = \tfrac{1}{2}(1 - \hat{\sigma}_t)^2 + \tfrac{1}{2}\hat{\sigma}_t^2 \;\geq\; \tfrac{1}{4},$$

where the inequality is minimized at $\hat{\sigma}_t = 1/2$.

Summing over all rounds. Taking total expectation and summing over $t = 1, \ldots, r$,

$$\mathbb{E}\left[\sum_{t=1}^{r} (\sigma_t - \hat{\sigma}_t)^2\right] \;\geq\; \frac{r}{4} \;=\; \frac{\lfloor \log_2 k \rfloor}{4}.$$

After r rounds, the candidate set S_{r+1} is reduced to a single element, the algorithm has identified the true index i^* and incurs no loss thereafter. Therefore, the expected cumulative squared loss is at least $\frac{1}{4}\lfloor \log_2 k \rfloor$, which equals the expected regret by Observation 1. Hence for any deterministic algorithm $\mathcal{A}$,

$$\mathbb{E}[\mathrm{Regret}_T(\mathcal{A})] \;\geq\; \tfrac{1}{4}\lfloor \log_2 k \rfloor = \Omega(\log k).$$

By Yao's minimax principle, the same lower bound holds for every randomized algorithm against worst-case inputs. □

The result matches the regret upper bound of $O(\log k)$, giving $\Theta(\log k)$ regret for range, subset, and linear queries.

C.5 Deferred Proofs for Section 4: Lower Bound under Absolute Loss

We provide the proofs of Lemmas 4.2 to 4.4.

Lemma 4.2. *For each round $t \in \{2, 3, \ldots, d\}$, the true selectivity of query $\mathbf{v}_t$ defined in* (3) *for $\tilde{\mathbf{w}}$ defined in* (2) *is $\sigma_t = \frac{1}{2}(1 + Y_t)$.*

Proof. The true selectivity is $\sigma_t = \langle \mathbf{v}_t, \tilde{\mathbf{w}} \rangle$. Since both $\mathbf{v}_t$ and $\tilde{\mathbf{w}}$ have zero in their last $k - d$ coordinates, the inner product reduces to the first d coordinates:

$$\sigma_t = \left\langle \frac{1}{2}(\mathbf{1} + \mathbf{h}_t),\ \frac{1}{d}\mathbf{H}^\top \mathbf{Y} \right\rangle = \frac{1}{2d}(\mathbf{1} + \mathbf{h}_t)^\top \mathbf{H}^\top \mathbf{Y} = \frac{1}{2d}\left(\mathbf{1}^\top \mathbf{H}^\top + \mathbf{h}_t^\top \mathbf{H}^\top\right)\mathbf{Y}. \tag{11}$$

By the orthogonality of normalized Hadamard matrices, we have $\mathbf{1}^\top \mathbf{H}^\top = \mathbf{h}_1^\top \mathbf{H}^\top = d\mathbf{e}_1^\top$ and $\mathbf{h}_t^\top \mathbf{H}^\top = d\mathbf{e}_t^\top$, where $\mathbf{e}_1, \mathbf{e}_t \in \{0,1\}^d$ denote the first and t-th standard basis vectors. Substituting these results back into (11), and recalling that $Y_1 = 1$:

$$\sigma_t = \frac{1}{2d}(d\mathbf{e}_1^\top + d\mathbf{e}_t^\top)\mathbf{Y} = \frac{1}{2}(Y_1 + Y_t) = \frac{1}{2}(1 + Y_t). \qquad \square$$

Lemma 4.3. *For each round $t \in \{2, \ldots, d\}$, let $\hat{\sigma}_t \in [0,1]$ be the deterministic algorithm's prediction given $Y_1, \ldots, Y_{t-1}$. Let $\mathit{Loss}_t(\tilde{\mathbf{w}}) = |\hat{\sigma}_t - \langle \mathbf{v}_t, \tilde{\mathbf{w}} \rangle|$ be the absolute loss at round t for $\tilde{\mathbf{w}}$. Then the expected loss over Y_t for $\tilde{\mathbf{w}}$ is given by:*

$$\mathbb{E}_{Y_t}[\mathit{Loss}_t(\tilde{\mathbf{w}}) \mid Y_1, \ldots, Y_{t-1}] \geq \frac{\delta}{2}.$$

Proof. Fix $(Y_1, \ldots, Y_{t-1})$. By Lemma 4.2, $\sigma_t = \frac{1}{2}(1 + Y_t)$, so the absolute loss is

$$\mathrm{Loss}_t(\tilde{\mathbf{w}}) = |\hat{\sigma}_t - \sigma_t| = \left|\hat{\sigma}_t - \frac{1}{2}(Y_t + 1)\right| = \frac{1}{2}\left|(2\hat{\sigma}_t - 1) - Y_t\right|.$$

Define $\hat{y}_t \coloneqq 2\hat{\sigma}_t - 1 \in [-1, 1]$, so that $\mathrm{Loss}_t(\tilde{\mathbf{w}}) = \frac{1}{2}|\hat{y}_t - Y_t|$. Since the algorithm is deterministic, $\hat{y}_t$ is a deterministic function of $(Y_1, \ldots, Y_{t-1})$, whereas $Y_t = \pm\delta$ with equal probability, independently of $(Y_1, \ldots, Y_{t-1})$. Therefore,

$$\mathbb{E}_{Y_t}[|\hat{y}_t - Y_t| \mid Y_1, \ldots, Y_{t-1}] = \frac{1}{2}|\delta - \hat{y}_t| + \frac{1}{2}|\hat{y}_t + \delta| \geq \delta, \tag{12}$$

where the inequality is the triangle inequality $|a| + |b| \geq |a + b|$ with $a = \delta - \hat{y}_t$ and $b = \hat{y}_t + \delta$. Hence $\mathbb{E}_{Y_t}[\mathrm{Loss}_t(\tilde{\mathbf{w}}) \mid Y_1, \ldots, Y_{t-1}] = \frac{1}{2}\mathbb{E}_{Y_t}[|\hat{y}_t - Y_t| \mid Y_1, \ldots, Y_{t-1}] \geq \delta/2$. $\square$

Lemma 4.4. *Let $E = \{\tilde{\mathbf{w}} \in \Delta_k\}$ denote the event that the candidate weight vector $\tilde{\mathbf{w}}$ defined in* (2) *lies in the simplex Δ_k. The probability that E holds is $\mathbb{P}(E) \geq 1 - 2d\exp\left(-\frac{1}{2d\delta^2}\right)$.*

Proof. For $\tilde{\mathbf{w}}$ to lie in the simplex Δ_k, it must satisfy (1) $\sum_{i=1}^k \tilde{w}_i = 1$ and (2) $\tilde{w}_i \geq 0$ for all $i \in [k]$. The first constraint holds for every realization of $\mathbf{Y}$, since $\mathbf{1}^\top \mathbf{H}^\top = d\,\mathbf{e}_1^\top$ and $Y_1 = 1$:

$$\sum_{i=1}^k \tilde{w}_i = \sum_{i=1}^d \tilde{w}_i = \frac{1}{d}\,\mathbf{1}^\top \mathbf{H}^\top \mathbf{Y} = \frac{1}{d} \cdot d\,\mathbf{e}_1^\top \mathbf{Y} = Y_1 = 1.$$

For the second constraint, the last $k - d$ coordinates of $\tilde{\mathbf{w}}$ are zero, so we focus on the first d coordinates. Each $\tilde{w}_i$ with $i \in [d]$ has the form $\tilde{w}_i = \frac{1}{d} + \frac{1}{d}\sum_{s=2}^d Y_s h_{s,i}$, where $h_{s,i}$ denotes the (s,i)-entry of $\mathbf{H}$ (the term $s = 1$ contributes $Y_1 h_{1,i}/d = 1/d$). Thus the non-negativity condition $\tilde{w}_i \geq 0$ is equivalent to $\sum_{s=2}^d Y_s h_{s,i} \geq -1$. Let $\tilde{E}$ denote the event that $|\sum_{s=2}^d Y_s h_{s,i}| \leq 1$ for all $i \in [d]$, i.e.,

$$\tilde{E} = \bigcap_{i=1}^d \left\{\left|\sum_{s=2}^d Y_s h_{s,i}\right| \leq 1\right\}.$$

If $\tilde{E}$ holds, then $\sum_{s=2}^d Y_s h_{s,i} \geq -1$ for all $i \in [d]$. Hence $\tilde{E} \subseteq E$, and it suffices to lower bound $\mathbb{P}(\tilde{E})$. We bound the probability of the complement event

$$\tilde{E}^c = \bigcup_{i=1}^d \left\{\left|\sum_{s=2}^d Y_s h_{s,i}\right| > 1\right\}.$$

Fix a coordinate $i \in [d]$. The terms $Y_s h_{s,i}$ for $s \in \{2, \ldots, d\}$ are independent random variables taking the values $-\delta$ and $+\delta$ with equal probability, so their sum has mean zero. Hoeffding's

inequality[2] gives

$$\mathbb{P}\Big(\Big|\sum_{s=2}^{d} Y_s h_{s,i}\Big| > 1\Big) \le 2\exp\Big(-\frac{2\cdot 1^2}{(d-1)(2\delta)^2}\Big) \le 2\exp\Big(-\frac{1}{2d\delta^2}\Big).$$

A union bound over the d coordinates gives $\mathbb{P}(\tilde{E}^c) \le 2d\exp\big(-\frac{1}{2d\delta^2}\big)$, and therefore

$$\mathbb{P}(E) \ge \mathbb{P}(\tilde{E}) = 1 - \mathbb{P}(\tilde{E}^c) \ge 1 - 2d\exp\Big(-\frac{1}{2d\delta^2}\Big). \qquad \square$$

C.6 Point Queries

Point queries on static databases admit constant regret under both squared and absolute loss (Table 2). Since each point query asks for exactly one coordinate of the fixed database vector, once a coordinate has been observed, its value is revealed. So all later queries to the same coordinate can be predicted with zero loss. This motivates the following Coordinate Memorization (CoordMem) algorithm (Algorithm 2). The algorithm maintains a database estimate vector $\hat{\mathbf{w}}$ and initializes it to $\mathbf{0}$. In each round t, it predicts the value of the queried coordinate i from the estimate $\hat{w}_i$, which is 0 if the coordinate is not yet queried, or the true weight w_i^* if it has been queried before. After observing the true selectivity $\sigma_t = w_i^*$, it memorizes the weight of i in the estimate.

Algorithm 2 Coordinate Memorization (CoordMem)

Initialize: $\hat{\mathbf{w}} \leftarrow \mathbf{0}$ ▷ Initialize the database estimate vector to the zero vector
for $t = 1, 2, \ldots, T$ **do**
  Receive point query $\mathbf{v}_t = \mathbf{e}_i$ ▷ the i-th standard basis vector
  Predict $\hat{\sigma}_t \leftarrow \hat{w}_i$
  Observe true selectivity $\sigma_t = w_i^*$ ▷ $\mathbf{w}^*$ is the true database vector
  Update $\hat{w}_i \leftarrow \sigma_t$ ▷ memorize coordinate i with its true value

The following proposition shows that CoordMem yields the tight constant-regret bounds under both absolute and squared loss.

Proposition C.5 (Point queries in static databases)**.** *In the static database setting, for point queries,* CoordMem *achieves regret at most* 1 *under both absolute and squared loss. Every (possibly randomized) online algorithm has worst-case expected regret at least* $\frac{1}{2}$ *under absolute loss and at least* $\frac{1}{4}$ *under squared loss.*

Proof. By Observation 1, in the static setting the regret of any algorithm equals its cumulative loss, so it suffices to bound the cumulative loss of the algorithm.

Upper bound. The CoordMem algorithm incurs loss only at the first time a coordinate is queried, where it predicts 0 while the true selectivity is w_i^*, incurring absolute loss w_i^* and squared loss $(w_i^*)^2$. Every subsequent query to coordinate i is predicted exactly and incurs zero loss. Summing over all coordinates that are ever queried, we obtain the following upper bounds on the cumulative absolute and squared loss:

$$\sum_{t=1}^{T} |\sigma_t - \hat{\sigma}_t| \;\le\; \sum_{i=1}^{k} w_i^* \;=\; 1, \qquad \sum_{t=1}^{T} (\sigma_t - \hat{\sigma}_t)^2 \;\le\; \sum_{i=1}^{k} (w_i^*)^2 \;\le\; \sum_{i=1}^{k} w_i^* \;=\; 1,$$

Therefore, the regret upper bound is 1 under both absolute and squared loss.

Lower bound. Before the game starts, the adversary chooses $\mathbf{w}^*$ uniformly from $\{\mathbf{e}_1, \mathbf{e}_2\} \subseteq \Delta_k$ and fixes the point query $\mathbf{v}_t = \mathbf{e}_1$ in every round. The remaining $k-2$ coordinates have zero mass. The first selectivity σ_1 is 0 or 1 with equal probability, independently of the algorithm's prediction $\hat{\sigma}_1$, including its internal randomness. Hence,

$$\mathbb{E}\big[|\sigma_1 - \hat{\sigma}_1|\big] = \mathbb{E}\big[\tfrac{1}{2}\hat{\sigma}_1 + \tfrac{1}{2}(1 - \hat{\sigma}_1)\big] = \tfrac{1}{2},$$

[2] We use the standard two-sided Hoeffding inequality (Hoeffding, 1963): for independent random variables $X_1, \ldots, X_n$ with $X_i \in [a_i, b_i]$ almost surely, $\mathbb{P}\big(\big|\sum_{i=1}^n X_i - \mathbb{E}\big[\sum_{i=1}^n X_i\big]\big| \ge \epsilon\big) \le 2\exp\Big(-\frac{2\epsilon^2}{\sum_{i=1}^n (b_i - a_i)^2}\Big)$ for any $\epsilon > 0$. The same bound applies to $\mathbb{P}(|\cdot| > \epsilon)$, since $\{|\cdot| > \epsilon\} \subseteq \{|\cdot| \ge \epsilon\}$.

$$\mathbb{E}\big[(\sigma_1 - \hat{\sigma}_1)^2\big] = \mathbb{E}\big[\tfrac{1}{2}\hat{\sigma}_1^2 + \tfrac{1}{2}(1 - \hat{\sigma}_1)^2\big] \geq \tfrac{1}{4}.$$

The losses in subsequent rounds are nonnegative. Thus, the expected cumulative loss under this distribution over fixed databases is at least $1/2$ under absolute loss and $1/4$ under squared loss. The worst-case expected regret is at least this average, since regret equals cumulative loss in the static setting. □

D Dynamic Databases under Squared Loss

D.1 Upper Bound: Exponentially Weighted Online Optimization

We show that the Exponentially Weighted Online Optimization (EWOO) algorithm (Hazan et al., 2007) achieves an $O(k \log T)$ regret bound for dynamic databases under squared loss, by exploiting the exp-concavity of the squared loss function, as shown in the following Lemma D.1. The algorithm is given in Algorithm 3, and the regret guarantee is stated in Corollary D.3.

In each round t, we view the selectivity prediction $\hat{\sigma}_t \in [0, 1]$ as arising from a predicted database vector $\hat{\mathbf{w}}_t \in \Delta_k$ via $\hat{\sigma}_t = \langle \mathbf{v}_t, \hat{\mathbf{w}}_t \rangle$. The round-$t$ squared loss, as a function of $\hat{\mathbf{w}}$, is then

$$\text{Loss}_t(\hat{\mathbf{w}}) := \big(\sigma_t - \langle \mathbf{v}_t, \hat{\mathbf{w}} \rangle\big)^2. \tag{13}$$

A convex function $f : \mathcal{K} \to \mathbb{R}$ is *α-exp-concave* over $\mathcal{K}$ if the function $g(\mathbf{x}) := \exp(-\alpha f(\mathbf{x}))$ is concave over $\mathcal{K}$ (Hazan, 2016). The squared loss in (13) satisfies this property, as the following lemma shows.

Lemma D.1. *For any $\sigma_t \in [0, 1]$ and $\mathbf{v}_t \in [0, 1]^k$, the function $\text{Loss}_t : \Delta_k \to \mathbb{R}$ defined in* (13) *is α-exp-concave over Δ_k with $\alpha = \frac{1}{2}$.*

Proof. Let $u(\hat{\mathbf{w}}) = \langle \mathbf{v}_t, \hat{\mathbf{w}} \rangle$. Because $\hat{\mathbf{w}} \in \Delta_k$ and $\mathbf{v}_t \in [0, 1]^k$, we have $u(\hat{\mathbf{w}}) \in [0, 1]$. Define the scalar function $g(u) = (\sigma_t - u)^2$ so that $\text{Loss}_t(\hat{\mathbf{w}}) = g(u(\hat{\mathbf{w}}))$. For twice-differentiable functions, α-exp-concavity is implied by the matrix inequality $\nabla^2 \text{Loss}_t(\hat{\mathbf{w}}) \succeq \alpha(\nabla \text{Loss}_t(\hat{\mathbf{w}}))(\nabla \text{Loss}_t(\hat{\mathbf{w}}))^\top$.

By the chain rule, $\nabla \text{Loss}_t(\hat{\mathbf{w}}) = g'(u)\mathbf{v}_t$ and $\nabla^2 \text{Loss}_t(\hat{\mathbf{w}}) = g''(u)\mathbf{v}_t\mathbf{v}_t^\top$. The required matrix inequality reduces to the scalar condition $g''(u) \geq \alpha(g'(u))^2$ because both sides are multiples of $\mathbf{v}_t\mathbf{v}_t^\top$. We compute $g'(u) = -2(\sigma_t - u)$ and $g''(u) = 2$, so $g''(u) \geq \alpha(g'(u))^2$ becomes

$$2 \geq 4\alpha(\sigma_t - u)^2.$$

Since $\sigma_t, u \in [0, 1]$, we have $(\sigma_t - u)^2 \leq 1$, and the inequality holds whenever $\alpha \leq \frac{1}{2}$. Choosing $\alpha = \frac{1}{2}$ satisfies the condition uniformly over Δ_k, establishing α-exp-concavity of Loss_t. □

By exploiting this exp-concavity, we achieve logarithmic regret via the Exponentially Weighted Online Optimization (EWOO) algorithm (Hazan et al., 2007); we follow the description given in Hazan (2016).

As shown in Algorithm 3, the decision space is the probability simplex, i.e., $\mathcal{K} = \Delta_k$. The parameter α is the exp-concavity parameter of the loss sequence. In each round t, EWOO: (i) computes a density function π_t over Δ_k in Line 2, which depends only on the cumulative loss up to round $t - 1$; (ii) predicts the database vector $\hat{\mathbf{w}}_t$ as the centroid (expected value) of π_t; (iii) outputs the selectivity prediction $\hat{\sigma}_t = \langle \mathbf{v}_t, \hat{\mathbf{w}}_t \rangle$; (iv) observes σ_t and incurs $\text{Loss}_t(\hat{\mathbf{w}}_t)$. Then for the next round $t+1$, each candidate $\mathbf{w}$ is reweighted by the factor $\exp(-\alpha \text{Loss}_t(\mathbf{w}))$, so candidates with larger squared loss receive smaller weight in future rounds. We restate the general regret bound from Hazan (2016) in Theorem D.2.

Theorem D.2 (General EWOO Regret Bound (Hazan, 2016))**.** *Let the loss functions $\text{Loss}_1, \ldots, \text{Loss}_T$ be α-exp-concave over a convex set $\mathcal{K} \subset \mathbb{R}^d$. The regret of the EWOO algorithm is bounded by:*

$$\textit{Regret}_T(\textit{EWOO}) \leq \frac{d}{\alpha} \log T + \frac{2}{\alpha}.$$

Since the dimension of our problem is k (the size of the support), and $\alpha = \frac{1}{2}$ by Lemma D.1, we obtain the following corollary:

Algorithm 3 Exponentially Weighted Online Optimization (EWOO) for selectivity estimation

Input: Convex set $\mathcal{K} = \Delta_k$, parameter $\alpha > 0$

1: **for** $t = 1, 2, \ldots, T$ **do**

2: **Compute the density function:** $\pi_t(\mathbf{w}) = \exp\big(-\alpha \sum_{\tau=1}^{t-1} \mathrm{Loss}_\tau(\mathbf{w})\big)$ for all $\mathbf{w} \in \Delta_k$

3: **Predict the database vector:** $\hat{\mathbf{w}}_t = \dfrac{\int_{\Delta_k} \mathbf{w} \cdot \pi_t(\mathbf{w})\, d\mathbf{w}}{\int_{\Delta_k} \pi_t(\mathbf{w})\, d\mathbf{w}}$

4: **Output the predicted selectivity:** $\hat{\sigma}_t = \langle \mathbf{v}_t, \hat{\mathbf{w}}_t \rangle$

5: **Observe and incur loss:** observe the true selectivity σ_t and incur $\mathrm{Loss}_t(\hat{\mathbf{w}}_t) = (\sigma_t - \langle \mathbf{v}_t, \hat{\mathbf{w}}_t \rangle)^2$

Corollary D.3. *For dynamic databases under the squared loss function, the EWOO algorithm (Algorithm 3) achieves the following regret bound for the general linear queries:*

$$\textit{Regret}_T(\mathrm{EWOO}) \le 2k \log T + 4 = O(k \log T).$$

Together with the $\Omega(\log T)$ lower bound in Theorem 2.2, this establishes the logarithmic regret rate as the optimal regret rate with respect to the time horizon T.

Polynomial-time algorithm with weaker regret bound: Online Newton Step algorithm While EWOO achieves the optimal regret rate, its exact implementation requires integrating over the simplex, which is computationally expensive. Another classical algorithm for exp-concave losses is the Online Newton Step (ONS) algorithm (Hazan, 2016), which maintains a single point estimate and updates it using a Hessian approximation. ONS runs in strictly polynomial time, at the cost of a slightly weaker dependence on k in the regret bound. We restate its general guarantee from Hazan (2016) below.

Theorem D.4 (General ONS Regret Bound (Hazan, 2016))**.** *Let $\mathit{Loss}_1, \ldots, \mathit{Loss}_T$ be α-exp-concave loss functions over a convex set $\mathcal{K} \subset \mathbb{R}^d$ with Euclidean diameter D, and suppose $\|\nabla \mathit{Loss}_t(\mathbf{w})\|_2 \le G$ for all $\mathbf{w} \in \mathcal{K}$ and $t \in [T]$. Then for $T \ge 4$, with parameters $\gamma = \frac{1}{2}\min\{\frac{1}{GD}, \alpha\}$ and $\varepsilon = \frac{1}{\gamma^2 D^2}$, the Online Newton Step algorithm satisfies*

$$\textit{Regret}_T(\mathrm{ONS}) \le 2\left(\frac{1}{\alpha} + GD\right) d \log T.$$

To instantiate this bound for our setting, we note that the simplex Δ_k has Euclidean diameter at most $D = \sqrt{2}$, and the squared-loss gradients satisfy $\|\nabla \mathrm{Loss}_t(\mathbf{w})\|_2 \le G = 2\sqrt{k}$ for all $\mathbf{w} \in \Delta_k$. Substituting $d = k$, $\alpha = \frac{1}{2}$ (Lemma D.1), $D = \sqrt{2}$, and $G = 2\sqrt{k}$ into Theorem D.4 gives the following corollary.

Corollary D.5. *For dynamic databases under squared loss, with general linear queries, the ONS algorithm achieves the following regret bound:*

$$\textit{Regret}_T(\mathrm{ONS}) \;\le\; 2\big(2 + 2\sqrt{2k}\big)\, k \log T = O(k^{3/2} \log T).$$

Thus ONS also achieves logarithmic regret in T, but with a slightly weaker dependence on k than EWOO. We use EWOO as our main tool because it yields the sharper $O(k \log T)$ regret bound.

D.2 Deferred Proofs for Section 5: Lower Bound under Squared Loss

We provide the proofs of Lemmas 5.1 to 5.3. In these proofs, we fix a deterministic algorithm and use the construction in Section 5 with $k = 2$.

Lemma 5.1. *For every deterministic algorithm, the expected regret satisfies $\mathbb{E}[\mathit{Regret}_T] \ge \sum_{t=1}^{T} \mathbb{E}\big[(\hat{\sigma}_t - \mu)^2\big]$.*

Proof. Since the query is fixed to $\mathbf{v}_t = (1, 0)$ for all t, any fixed database vector $\mathbf{u} = (u_1, u_2) \in \Delta_2$ yields a constant prediction $\langle \mathbf{v}_t, \mathbf{u} \rangle = u_1$ in every round. Thus, comparing against the best fixed database vector in (1) is equivalent to comparing against the best constant prediction. Furthermore, since $(\mu, 1 - \mu) \in \Delta_2$ is a valid fixed database, the loss of the best fixed comparator is at most

$\sum_{t=1}^{T}(\sigma_t-\mu)^2$. Therefore,

$$\mathbb{E}[\text{Regret}_T] = \mathbb{E}\left[\sum_{t=1}^{T}(\hat{\sigma}_t-\sigma_t)^2 - \min_{x\in[0,1]}\sum_{t=1}^{T}(\sigma_t-x)^2\right]$$
$$\geq \sum_{t=1}^{T}\mathbb{E}\left[(\hat{\sigma}_t-\sigma_t)^2-(\sigma_t-\mu)^2\right].$$

Fix a round t. Since the algorithm is deterministic, its prediction $\hat{\sigma}_t$ is determined by $\mathcal{H}_t = (\mathbf{v}_1,\sigma_1,\ldots,\mathbf{v}_{t-1},\sigma_{t-1},\mathbf{v}_t)$, we have $\hat{\sigma}_t \mid \mathcal{H}_t,\mu = \hat{\sigma}_t$. Similarly, σ_t depends only on μ. Hence, $\mathbb{E}[\sigma_t \mid \mathcal{H}_t,\mu]=\mu$. We condition on μ and $\mathcal{H}_t$, and write:

$$\mathbb{E}\left[(\hat{\sigma}_t-\sigma_t)^2-(\sigma_t-\mu)^2 \mid \mathcal{H}_t,\mu\right]$$
$$= (\hat{\sigma}_t-\mu)\,\mathbb{E}\left[\hat{\sigma}_t+\mu-2\sigma_t \mid \mathcal{H}_t,\mu\right]$$
$$= (\hat{\sigma}_t-\mu)(\hat{\sigma}_t+\mu-2\mu) = (\hat{\sigma}_t-\mu)^2.$$

Taking the expectation over $\mathcal{H}_t$ and μ and summing over t gives

$$\mathbb{E}[\text{Regret}_T] \geq \sum_{t=1}^{T}\mathbb{E}\left[(\hat{\sigma}_t-\mu)^2\right]. \qquad \square$$

Lemma 5.2. *For every round t and every deterministic algorithm, $\mathbb{E}\big[(\hat{\sigma}_t-\mu)^2\big] \geq \mathbb{E}\big[\mathrm{Var}(\mu \mid \mathcal{H}_t)\big]$.*

Proof. We find the prediction $\hat{\sigma}_t$ that minimizes $\mathbb{E}[(\hat{\sigma}_t-\mu)^2 \mid \mathcal{H}_t]$. Since the algorithm is deterministic, $\hat{\sigma}_t$ is fixed given $\mathcal{H}_t$, and

$$\mathbb{E}\left[(\hat{\sigma}_t-\mu)^2 \mid \mathcal{H}_t\right] = \hat{\sigma}_t^2 - 2\hat{\sigma}_t\mathbb{E}[\mu \mid \mathcal{H}_t] + \mathbb{E}[\mu^2 \mid \mathcal{H}_t].$$

Taking the derivative with respect to $\hat{\sigma}_t$ and setting it to zero gives

$$\frac{d}{d\hat{\sigma}_t}\mathbb{E}\left[(\hat{\sigma}_t-\mu)^2 \mid \mathcal{H}_t\right] = 2\hat{\sigma}_t - 2\mathbb{E}[\mu \mid \mathcal{H}_t] = 0.$$

The second derivative is $2>0$. Thus, the minimizing prediction is the posterior mean $\mathbb{E}[\mu \mid \mathcal{H}_t] \in [0,1]$, and the minimum conditional expected squared error is $\mathrm{Var}(\mu \mid \mathcal{H}_t)$. Therefore,

$$\mathbb{E}\left[(\hat{\sigma}_t-\mu)^2 \mid \mathcal{H}_t\right] \geq \mathrm{Var}(\mu \mid \mathcal{H}_t).$$

Taking the expectation over $\mathcal{H}_t$ gives the claim. $\square$

Lemma 5.3. *For every round t, $\mathbb{E}\big[\mathrm{Var}(\mu \mid \mathcal{H}_t)\big] = \frac{1}{6(t+1)}$.*

Proof. The uniform distribution on $[0,1]$ is the $\mathrm{Beta}(1,1)$ distribution. By standard properties of the Beta distribution (Johnson et al., 2022; Berger, 2013), after observing $t-1$ outcomes $\sigma_1,\ldots,\sigma_{t-1}$ that are independent $\mathrm{Bernoulli}(\mu)$ draws conditional on μ, the posterior distribution is

$$\mu \mid \mathcal{H}_t \sim \mathrm{Beta}\big(1+H_{t-1},\, t-H_{t-1}\big),$$

where $H_{t-1}=\sum_{s=1}^{t-1}\sigma_s$ is the number of rounds in which the database is $(1,0)$ among the first $t-1$ rounds.

The variance of $\mathrm{Beta}(\alpha,\beta)$ is $\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$. With $\alpha = 1+H_{t-1}$ and $\beta = t-H_{t-1}$, we obtain

$$\mathrm{Var}(\mu \mid \mathcal{H}_t) = \frac{(1+H_{t-1})(t-H_{t-1})}{(t+1)^2(t+2)}.$$

Since $H_{t-1} \mid \mu \sim \mathrm{Binomial}(t-1,\mu)$ and $\mu \sim \mathrm{Uniform}[0,1]$, for every $\zeta \in \{0,1,\ldots,t-1\}$,

$$\Pr(H_{t-1}=\zeta) = \int_0^1 \binom{t-1}{\zeta}\mu^{\zeta}(1-\mu)^{t-1-\zeta}\,d\mu$$
$$= \binom{t-1}{\zeta}\frac{\zeta!\,(t-1-\zeta)!}{t!} = \frac{1}{t}.$$

Therefore,

$$\mathbb{E}\left[\mathrm{Var}(\mu \mid \mathcal{H}_t)\right] = \frac{1}{t(t+1)^2(t+2)} \sum_{\zeta=0}^{t-1} (1+\zeta)(t-\zeta)$$
$$= \frac{1}{t(t+1)^2(t+2)} \sum_{\zeta=0}^{t-1} \left(t + (t-1)\zeta - \zeta^2\right).$$

Using $\sum_{\zeta=0}^{t-1} \zeta = \frac{(t-1)t}{2}$ and $\sum_{\zeta=0}^{t-1} \zeta^2 = \frac{(t-1)t(2t-1)}{6}$, we get

$$\mathbb{E}\left[\mathrm{Var}(\mu \mid \mathcal{H}_t)\right] = \frac{1}{t(t+1)^2(t+2)} \cdot \frac{t(t+1)(t+2)}{6} = \frac{1}{6(t+1)}. \qquad \square$$

The construction in Section 5 uses a single fixed point query, so the $\Omega(\log T)$ lower bound holds for point queries and hence, by the nesting of the query classes, for range, subset and linear queries.

E Dynamic Databases under Absolute Loss

In this section, we prove the regret bounds under absolute loss for dynamic databases in Theorem 2.2, and we give the finer bounds for point queries. Section E.1 establishes the lower bounds: Proposition E.1 (Section E.1.1) gives $\Omega(\sqrt{T})$ for point queries, and Section E.1.2 proves the $\Omega(\sqrt{T \log k})$ lower bound for subset queries. Section E.2 establishes the upper bounds with the Follow-the-Regularized-Leader (FTRL) algorithm (Shalev-Shwartz, 2012): with squared-ℓ_2 regularization it achieves $O(\sqrt{T})$ for point queries (Corollary E.9, Section E.2.1), and with negative-entropy regularization it achieves $O(\sqrt{T \log k})$ for general linear queries (Corollary E.13, Section E.2.2). Therefore, the regret for point queries is $\Theta(\sqrt{T})$, independent of the support size k, whereas for subset and general linear queries it is $\Theta(\sqrt{T \log k})$.

E.1 Lower Bound

E.1.1 Point Queries

Proposition E.1. *For dynamic databases under the absolute loss function, for point queries, any (randomized) online algorithm incurs an (expected) regret of $\Omega(\sqrt{T})$ in the worst case.*

Proof. It suffices to consider $k = 2$. The adversary fixes the point query $\mathbf{v}_t = (1, 0)$ for all $t \in [T]$. Independently in each round, it sets $\mathbf{w}_t = (1, 0)$ or $\mathbf{w}_t = (0, 1)$ with equal probability. Hence the true selectivity $\sigma_t = \langle \mathbf{v}_t, \mathbf{w}_t \rangle \sim \mathrm{Bernoulli}(1/2)$, taking values 1 or 0 with equal probability.

The algorithm's prediction $\hat{\sigma}_t \in [0, 1]$ depends only on the history $\mathcal{H}_t = \{\mathbf{v}_1, \sigma_1, \mathbf{v}_2, \sigma_2, \ldots, \mathbf{v}_{t-1}, \sigma_{t-1}, \mathbf{v}_t\}$, and is independent of $\mathbf{w}_t$. Since $\mathbf{v}_t$ is fixed, this implies that $\hat{\sigma}_t$ is independent of $\sigma_t = \langle \mathbf{v}_t, \mathbf{w}_t \rangle$. Hence the per-round expected loss is

$$\mathbb{E}[\mathrm{Loss}_t] = \mathbb{E}\big[|\sigma_t - \hat{\sigma}_t|\big] = \tfrac{1}{2}(1 - \hat{\sigma}_t) + \tfrac{1}{2}\hat{\sigma}_t = \tfrac{1}{2},$$

and by linearity, the expected cumulative loss over T rounds is $\mathbb{E}\big[\sum_{t=1}^{T} \mathrm{Loss}_t\big] = \frac{T}{2}$.

Then we derive an upper bound of the expected total loss of the best fixed predictor. Since the query is fixed, any $\mathbf{w} \in \Delta_2$ yields a constant prediction $c = w_1 \in [0, 1]$. Let $H = \sum_{t=1}^{T} \sigma_t$ count the rounds where $\sigma_t = 1$ (so $T - H$ counts the rounds where $\sigma_t = 0$), then the total loss can be written as:

$$\min_{\mathbf{w} \in \Delta_k} \sum_{t=1}^{T} \mathrm{Loss}_t(\mathbf{w}) = \min_{c \in [0,1]} \sum_{t=1}^{T} |\sigma_t - c|$$
$$= H(1-c) + (T-H)c = H + (\frac{T}{2} - H) \cdot 2c$$

Choosing $c = 0$ when $H \leq T/2$ and $c = 1$ otherwise gives:

$$\min_{\mathbf{w}\in\Delta_k} \sum_{t=1}^{T} \text{Loss}_t(\mathbf{w}) \leq \min(H, T-H) = \frac{T}{2} - \frac{|2H-T|}{2}$$

The last step uses the identity: $\min(a,b) = \frac{a+b}{2} - \frac{|a-b|}{2}, \forall a, b \in \mathbb{R}$.

Therefore, fixing an algorithm, the expected regret is:

$$\mathbb{E}\left[\sum_{t=1}^{T} \text{Loss}_t - \min_{\mathbf{w}\in\Delta_2} \sum_{t=1}^{T} \text{Loss}_t(\mathbf{w})\right] \geq \mathbb{E}\left[\frac{T}{2} - \frac{T}{2} + \frac{|2H-T|}{2}\right] = \mathbb{E}\left[|H - \tfrac{T}{2}|\right]$$

Since $\sigma_t \sim \text{Bernoulli}(1/2)$, H follows a binomial distribution $B(T, 1/2)$. By standard properties of the mean absolute deviation of the binomial distribution, we have $\mathbb{E}\left[|H - T/2|\right] \geq \sqrt{\frac{Var(H)}{2}} = \frac{\sqrt{T}}{2\sqrt{2}}$ where $T \geq 2$ (Berend & Kontorovich, 2013). Therefore, the expected regret of any algorithm against this randomized adversary is $\Omega(\sqrt{T})$.

Moreover, for any $k > 2$, the lower bound can be extended by embedding the same construction in the first two coordinates, with zero mass assigned to all remaining coordinates. Since the worst-case regret over adversary strategies is at least the expected regret under any specific adversary distribution, the lower bound holds. □

E.1.2 Subset Queries (Theorem 2.2)

We prove the $\Omega(\sqrt{T \log k})$ lower bound under absolute loss in Theorem 2.2, which holds even for subset queries.

Overview of Techniques. In each round $t \in [T]$, the adversary draws the true selectivity σ_t and the subset query $\mathbf{v}_t$ from independent distributions: $\sigma_t \sim \text{Bernoulli}(1/2)$, and $\mathbf{v}_t \in \{0,1\}^k$ has its first $k-1$ coordinates drawn i.i.d. from $\text{Bernoulli}(1/2)$. The last coordinate is set to $v_{t,k} = 1 - v_{t,1}$ to ensure that $\mathbf{v}_t$ is never the all-zero or all-one vector. Therefore, there always exists a valid database vector $\mathbf{w}_t$ that places all its mass on a single coordinate $j \in [k]$ satisfying $v_{t,j} = \sigma_t$.

Under this distribution, the regret reduces to the expected maximum of $k-1$ Rademacher random walks of length T. Since σ_t is independent of the algorithm's prediction, the algorithm incurs expected per-round loss $1/2$, giving total expected loss $T/2$. The best fixed comparator, however, concentrates on one of the k vertices of Δ_k, achieving total loss: $\min_{i\in[k]} \sum_t |\sigma_t - v_{t,i}|$. Defining $z_{t,i} := 1 - 2|\sigma_t - v_{t,i}|$, the expected regret becomes $\frac{1}{2}\,\mathbb{E}\big[\max_{i\in[k]} \sum_t z_{t,i}\big]$. By construction, the variables $\{z_{t,i}\}_{t\in[T],\, i\in[k-1]}$ are i.i.d. Rademacher random variables taking values in $\{-1,+1\}$ with probability $1/2$; the expected maximum of $k-1$ such sums of length T is $\Omega(\sqrt{T\log k})$ by standard results in (Cesa-Bianchi & Lugosi, 2006) (see Lemma E.4). By Yao's minimax principle, the lower bound on the regret of any randomized algorithm against the worst-case input is $\Omega(\sqrt{T\log k})$.

Proof of the absolute-loss lower bound in Theorem 2.2. **Step 1: Adversary construction.** At each round $t \in [T]$, the adversary first draws $k-1$ independent random variables from $\text{Bernoulli}(1/2)$: $b_{t,1}, \ldots, b_{t,k-1}$. Then it defines the subset query $\mathbf{v}_t \in \{0,1\}^k$ by:

$$v_{t,i} = b_{t,i} \quad \text{for } i = 1, \ldots, k-1, \qquad v_{t,k} = 1 - b_{t,1}.$$

Since the first and last coordinates always sum to 1, $\mathbf{v}_t$ is never equal to $\mathbf{0}$ or $\mathbf{1}$. Then the adversary draws a true selectivity independent of $\mathbf{v}_t$ and all past variables:

$$\sigma_t \sim \text{Bernoulli}(1/2).$$

Then it sets the database state $\mathbf{w}_t = \mathbf{e}_j$, where j is any coordinate satisfying $v_{t,j} = \sigma_t$. Since $\mathbf{v}_t$ contains at least one coordinate equal to 1 and at least one coordinate equal to 0, such a j always exists for either value of σ_t. This ensures $\langle \mathbf{v}_t, \mathbf{w}_t \rangle = v_{t,j} = \sigma_t$ and $\mathbf{w}_t$ is always a valid database state.

Step 2: Algorithm's expected loss. The learner observes $\mathbf{v}_t$ and then predicts $\hat{\sigma}_t = \langle \mathbf{v}_t, \hat{\mathbf{w}}_t \rangle \in [0,1]$. Since σ_t is drawn independent of $\hat{\sigma}_t$, the expected loss per-round is:

$$\mathbb{E}[\text{Loss}_t(\hat{\mathbf{w}}_t)] = \mathbb{E}\left[|\sigma_t - \hat{\sigma}_t|\right] = \tfrac{1}{2}\hat{\sigma}_t + \tfrac{1}{2}(1 - \hat{\sigma}_t) = \tfrac{1}{2}.$$

By linearity of expectation, the expected total loss for any algorithm is:

$$\mathbb{E}\big[\sum_{t=1}^{T}\text{Loss}_t(\hat{\mathbf{w}}_t)\big] = \frac{T}{2}. \tag{14}$$

Step 3: Best fixed predictor's expected loss. Consider any fixed weight vector $\mathbf{w}^* \in \Delta_k$. Its absolute loss in round t is $\text{Loss}_t(\mathbf{w}^*) = |\sigma_t - \langle \mathbf{v}_t, \mathbf{w}^*\rangle|$. Since coordinates of $\mathbf{w}^*$ sum to 1, we can rewrite the scalar σ_t as:

$$\sigma_t = \sum_{i=1}^{k} w_i^* \sigma_t.$$

Substituting this and the inner product $\langle \mathbf{v}_t, \mathbf{w}^*\rangle = \sum_{i=1}^{k} w_i^* v_{t,i}$ into the absolute loss yields:

$$\text{Loss}_t(\mathbf{w}^*) = \left|\sum_{i=1}^{k} w_i^*\sigma_t - \sum_{i=1}^{k} w_i^* v_{t,i}\right| = \left|\sum_{i=1}^{k} w_i^*(\sigma_t - v_{t,i})\right|.$$

Since $\sigma_t, v_{t,i} \in \{0,1\}$, the differences $(\sigma_t - v_{t,i})$ are non-negative for all $i \in [k]$ when $\sigma_t = 1$, or non-positive when $\sigma_t = 0$. Because $w_i^* \geq 0$, all terms $w_i^*(\sigma_t - v_{t,i})$ share the same sign across all $i \in [k]$, so the absolute value of the sum equals the sum of the absolute values. Rewriting the loss at round t as:

$$\text{Loss}_t(\mathbf{w}^*) = \sum_{i=1}^{k} w_i^* |\sigma_t - v_{t,i}|.$$

Consequently, the total loss of a fixed $\mathbf{w}^*$ can be written as follows:

$$\sum_{t=1}^{T}\text{Loss}_t(\mathbf{w}^*) = \sum_{i=1}^{k} w_i^* \left(\sum_{t=1}^{T} |\sigma_t - v_{t,i}|\right).$$

Define $S_i := \sum_{t=1}^{T} |\sigma_t - v_{t,i}|$ for each $i \in [k]$. The total loss is the linear function $\sum_{i=1}^{k} w_i^* S_i$ over $\mathbf{w}^* \in \Delta_k$. A linear function on the simplex attains its minimum at a vertex, so:

$$\min_{\mathbf{w}^*\in\Delta_k} \sum_{t=1}^{T}\text{Loss}_t(\mathbf{w}^*) = \min_{i\in[k]} S_i = \min_{i\in[k]} \sum_{t=1}^{T} |\sigma_t - v_{t,i}|. \tag{15}$$

Then we rewrite the total loss by defining $z_{t,i} := 1 - 2|\sigma_t - v_{t,i}|$. Substituting $|\sigma_t - v_{t,i}| = \frac{1}{2}(1 - z_{t,i})$ into (15) and taking expectations gives the expected total loss of the best fixed predictor:

$$\begin{aligned} &\mathbb{E}\left[\min_{\mathbf{w}^*\in\Delta_k} \sum_{t=1}^{T}\text{Loss}_t(\mathbf{w}^*)\right] = \mathbb{E}\left[\min_{i\in[k]} \sum_{t=1}^{T} |\sigma_t - v_{t,i}|\right] \\ =&\mathbb{E}\left[\min_{i\in[k]} \sum_{t=1}^{T} \frac{1 - z_{t,i}}{2}\right] = \frac{T}{2} - \frac{1}{2}\mathbb{E}\left[\max_{i\in[k]} \sum_{t=1}^{T} z_{t,i}\right]. \end{aligned} \tag{16}$$

Step 4: Reducing the expected regret to independent Rademacher sums. Combining (16) with (14) and since the maximum over all k coordinates is at least the maximum over the first $k-1$ coordinates, the expected regret for any algorithm is:

$$\mathbb{E}\left[\sum_{t=1}^{T}\text{Loss}_t(\hat{\mathbf{w}}_t)\right] - \mathbb{E}\left[\min_{\mathbf{w}^*\in\Delta_k} \sum_{t=1}^{T}\text{Loss}_t(\mathbf{w}^*)\right] = \frac{1}{2}\,\mathbb{E}\left[\max_{i\in[k]} \sum_{t=1}^{T} z_{t,i}\right] \geq \frac{1}{2}\mathbb{E}\left[\max_{i\in[k-1]} \sum_{t=1}^{T} z_{t,i}\right].$$

Recall that for each coordinate $i \in [k-1]$ and each round $t \in [T]$, both σ_t and $v_{t,i}$ are independent random variables drawn from $\text{Bernoulli}(1/2)$. Therefore, $z_{t,i} = 1 - 2|\sigma_t - v_{t,i}|$ is independent across $i \in [k-1]$ and $t \in [T]$, and $z_{t,i} \in \{-1, +1\}$ each with probability $1/2$, i.e., a Rademacher random variable. By the standard result as shown in the following Lemma E.4 (Cesa-Bianchi & Lugosi, 2006), the expected maximum of the sum of i.i.d. Rademacher random variables is $\Omega(\sqrt{T\log k})$. Therefore, for every deterministic algorithm, the expected regret under this randomized adversary

is $\Omega(\sqrt{T \log k})$ for sufficiently large T and k. By Yao's minimax principle, this implies the same lower bound on the worst-case expected regret of every randomized algorithm. □

The proof relies on two standard results from (Cesa-Bianchi & Lugosi, 2006), restated below.

Lemma E.2 (Cesa-Bianchi & Lugosi, 2006, Lemma A.11)**.** *Let $\{Z_{i,t}\}_{i\in[N],\, t\geq 1}$ be i.i.d. Rademacher random variables (i.e., $\mathbb{P}(Z_{i,t} = -1) = \mathbb{P}(Z_{i,t} = 1) = 1/2$), and let $G_1, \ldots, G_N$ be independent standard normal random variables. Then*

$$\lim_{n\to\infty} \mathbb{E}\left[\max_{i\in[N]} \frac{1}{\sqrt{n}} \sum_{t=1}^{n} Z_{i,t}\right] = \mathbb{E}\left[\max_{i\in[N]} G_i\right].$$

Lemma E.3 (Cesa-Bianchi & Lugosi, 2006, Lemma A.12)**.** *Let $G_1, \ldots, G_N$ be independent standard normal random variables. Then*

$$\lim_{N\to\infty} \frac{\mathbb{E}[\max_{i\in[N]} G_i]}{\sqrt{2 \ln N}} = 1.$$

Lemma E.4 (Expected maximum of Rademacher sums (Cesa-Bianchi & Lugosi, 2006))**.** *Let $\{z_{t,i}\}_{t\in[T],i\in[k-1]}$ be i.i.d. Rademacher random variables (i.e., $z_{t,i} \in \{-1, +1\}$ each with probability $1/2$). Then for sufficiently large T and k,*

$$\mathbb{E}\Big[\max_{i\in[k-1]} \sum_{t=1}^{T} z_{t,i}\Big] = \Omega(\sqrt{T \log k}).$$

Proof. First, by the central limit theorem (Lemma E.2), $\sum_{t=1}^{T} z_{t,i}/\sqrt{T}$ converges in distribution to a standard normal G_i as $T \to \infty$:

$$\lim_{T\to\infty} \mathbb{E}\Big[\max_{i\in[k-1]} \frac{1}{\sqrt{T}} \sum_{t=1}^{T} z_{t,i}\Big] = \mathbb{E}\Big[\max_{i\in[k-1]} G_i\Big],$$

where $G_1, \ldots, G_{k-1}$ are i.i.d. standard normal random variables. Second, the expected maximum of $k-1$ i.i.d. standard normal random variables satisfies (Lemma E.3):

$$\lim_{k\to\infty} \frac{\mathbb{E}\big[\max_{i\in[k-1]} G_i\big]}{\sqrt{2\ln(k-1)}} = 1.$$

Combining them and multiplying by $\sqrt{T}$, we have that for sufficiently large T and k:

$$\mathbb{E}\Big[\max_{i\in[k-1]} \sum_{t=1}^{T} z_{t,i}\Big] = \Omega(\sqrt{T \log k}).$$

□

E.2 Upper Bound: Follow-the-Regularized-Leader (FTRL) Algorithm

The choice of regularization function is dictated by the geometry of the query vectors. Point queries are standard basis vectors and satisfy the strong condition $\|\mathbf{v}_t\|_2 = 1$, which lets us use a squared ℓ_2 regularization function and obtain $O(\sqrt{T})$ regret independent of k. Linear queries only satisfy the weaker $\|\mathbf{v}_t\|_\infty \leq 1$ (with $\|\mathbf{v}_t\|_2$ as large as $\sqrt{k}$), for which the matching choice is the negative entropy regularization function, yielding $O(\sqrt{T \log k})$ regret.

Both algorithms follow the generic FTRL procedure as detailed in Algorithm 4: they maintain an estimated database vector $\hat{\mathbf{w}}_t$ in each round, by solving an optimization problem over the cumulative loss and a regularization term. The regularization function $R(\mathbf{w})$ prevents $\hat{\mathbf{w}}_t$ from changing too drastically between rounds and ensures stability. Then the algorithm outputs the estimated selectivity $\hat{\sigma}_t = \langle \mathbf{v}_t, \hat{\mathbf{w}}_t \rangle$ based on the estimated database vector.

The standard result of FTRL (Shalev-Shwartz, 2012) gives an upper bound on the regret relative to an arbitrary comparator $\mathbf{u}$, restated below:

Theorem E.5 (General FTRL Regret Bound (Shalev-Shwartz, 2012))**.** *Consider a sequence of convex loss functions $\ell_1, \ldots, \ell_T$, such that each ℓ_t is L_t-Lipschitz with respect to some norm $\|\cdot\|$. Let L satisfy $\frac{1}{T}\sum_{t=1}^{T} L_t^2 \leq L^2$. Let the regularization function R be a σ-strongly convex function with*

Algorithm 4 FTRL for Dynamic Databases under Absolute Loss

Input: Learning rate $\eta > 0$, regularization function $R(\mathbf{w})$: $\frac{1}{\eta}\sum_{i=1}^{k} w_i \log w_i$ for linear queries or $\frac{1}{\eta}\sum_{i=1}^{k} w_i^2$ for point queries

1: Initialize the estimated database vector: $\widehat{\mathbf{w}}_1 \leftarrow (1/k, \ldots, 1/k)$ (uniform distribution)
2: **for** $t = 1, 2, \ldots, T$ **do**
3: **Predict selectivity:** receive query $\mathbf{v}_t$ and output: $\hat{\sigma}_t \leftarrow \langle \mathbf{v}_t, \widehat{\mathbf{w}}_t \rangle$
4: **Compute loss:** receive true selectivity σ_t and incur $\text{Loss}_t = |\sigma_t - \hat{\sigma}_t|$
5: **Update the estimated database vector:** $\widehat{\mathbf{w}}_{t+1} \leftarrow \arg\min_{\mathbf{w}\in\Delta_k} \Big[\sum_{\tau=1}^{t} \text{Loss}_\tau(\mathbf{w}) + R(\mathbf{w})\Big]$, where $\text{Loss}_\tau(\mathbf{w}) = |\sigma_\tau - \langle \mathbf{v}_\tau, \mathbf{w}\rangle|$ denotes the absolute loss incurred by $\mathbf{w}$ on round τ.

respect to the same norm $\|\cdot\|$ over a convex set $\mathcal{K}$. $\textit{Regret}_T(\mathbf{u}) = \sum_{t=1}^{T} \textit{Loss}_t(\hat{\mathbf{w}}_t) - \sum_{t=1}^{T} \textit{Loss}_t(\mathbf{u})$ is the regret relative to the comparator $\mathbf{u}$. Then, for all $\mathbf{u} \in \mathcal{K}$,

$$\textit{Regret}_T(\mathbf{u}) = \sum_{t=1}^{T} \textit{Loss}_t(\hat{\mathbf{w}}_t) - \sum_{t=1}^{T} \textit{Loss}_t(\mathbf{u}) \le R(\mathbf{u}) - \min_{\mathbf{v}\in\mathcal{K}} R(\mathbf{v}) + \frac{TL^2}{\sigma},$$

To derive the specific regret bound for each query class, we establish three key properties separately in Section E.2.1 for point queries and Section E.2.2 for linear queries: (1) the strong convexity of the regularization function, (2) the Lipschitz continuity of the loss function, and (3) the diameter of the regularization function over the simplex.

E.2.1 Point Queries

For point queries we instantiate Algorithm 4 with the squared ℓ_2 regularizer

$$R(\mathbf{w}) \;=\; \tfrac{1}{\eta}\,\|\mathbf{w}\|_2^2 \;=\; \tfrac{1}{\eta}\sum_{i=1}^{k} w_i^2,$$

where $\eta > 0$ is the learning rate. The following three lemmas instantiate Theorem E.5, all measured in the ℓ_2 norm.

Lemma E.6. *$R(\mathbf{w}) = \frac{1}{\eta}\|\mathbf{w}\|_2^2$ is $\frac{2}{\eta}$-strongly convex with respect to the ℓ_2 norm on Δ_k.*

Proof. For a differentiable convex f, f is σ-strongly convex w.r.t. a norm $\|\cdot\|_2$ iff its Bregman divergence satisfies, for all $\mathbf{p}, \mathbf{q}$,

$$D_f(\mathbf{p}, \mathbf{q}) := f(\mathbf{p}) - f(\mathbf{q}) - \langle \nabla f(\mathbf{q}), \mathbf{p} - \mathbf{q}\rangle \;\ge\; \frac{\sigma}{2}\,\|\mathbf{p} - \mathbf{q}\|_2^2.$$

For $f(\mathbf{w}) = \|\mathbf{w}\|_2^2 = \sum_{i=1}^{k} w_i^2$, we have $\nabla f(\mathbf{w}) = 2\mathbf{w}$. Thus for any $\mathbf{p}, \mathbf{q} \in \Delta_k$:

$$\begin{aligned} D_f(\mathbf{p}, \mathbf{q}) &= \|\mathbf{p}\|_2^2 - \|\mathbf{q}\|_2^2 - \langle 2\mathbf{q},\, \mathbf{p} - \mathbf{q}\rangle \\ &= \sum_i p_i^2 + \sum_i q_i^2 - 2\sum_i p_i q_i \\ &= \|\mathbf{p} - \mathbf{q}\|_2^2 \;\ge\; \frac{2}{2}\,\|\mathbf{p} - \mathbf{q}\|_2^2, \end{aligned}$$

which is exactly the 2-strong convexity of f w.r.t. $\|\cdot\|_2$.

Finally, strong convexity scales linearly with positive constants. For $g = \alpha \cdot f$ with $\alpha > 0$, if f is σ-strongly convex, then:

$$\begin{aligned} D_g(\mathbf{p}, \mathbf{q}) &= \alpha f(\mathbf{p}) - \alpha f(\mathbf{q}) - \langle \alpha \nabla f(\mathbf{q}), \mathbf{p} - \mathbf{q}\rangle \\ &\ge \alpha \frac{\sigma}{2}\,\|\mathbf{p} - \mathbf{q}\|^2, \end{aligned}$$

Thus g is $\alpha\sigma$-strongly convex. Applying this to our case, we get that $R(\mathbf{w}) = \frac{1}{\eta}\cdot\|\mathbf{w}\|_2^2$ is $\frac{2}{\eta}$-strongly convex w.r.t. $\|\cdot\|_2$. □

Lemma E.7. *For any $\sigma_t \in [0,1]$ and any point query $\mathbf{v}_t = \mathbf{e}_{i_t}$, the loss function $\text{Loss}_t(\mathbf{w}) = |\sigma_t - \langle \mathbf{v}_t, \mathbf{w}\rangle|$ is 1-Lipschitz with respect to the ℓ_2 norm on Δ_k.*

Proof. For any $\mathbf{w}, \mathbf{w}' \in \Delta_k$:

$$\begin{aligned}|\text{Loss}_t(\mathbf{w}) - \text{Loss}_t(\mathbf{w}')| &= ||\sigma_t - \langle \mathbf{v}_t, \mathbf{w}\rangle| - |\sigma_t - \langle \mathbf{v}_t, \mathbf{w}'\rangle|| \\ &\le |\langle \mathbf{v}_t, \mathbf{w} - \mathbf{w}'\rangle| \quad \text{(triangle inequality)} \\ &\le \|\mathbf{v}_t\|_2 \cdot \|\mathbf{w} - \mathbf{w}'\|_2 \quad \text{(Cauchy–Schwarz inequality).}\end{aligned}$$

For point queries, $\mathbf{v}_t = \mathbf{e}_{i_t}$, so $\|\mathbf{v}_t\|_2 = 1$. Therefore:

$$|\text{Loss}_t(\mathbf{w}) - \text{Loss}_t(\mathbf{w}')| \le \|\mathbf{w} - \mathbf{w}'\|_2.$$

This establishes that the loss is 1-Lipschitz with respect to the ℓ_2 norm. □

Lemma E.8. *For $R(\mathbf{w}) = \frac{1}{\eta}\|\mathbf{w}\|_2^2$ on Δ_k and any $\mathbf{u} \in \Delta_k$, $R(\mathbf{u}) - \min_{\mathbf{w}\in\Delta_k} R(\mathbf{w}) \le \frac{1}{\eta}$.*

Proof. We first determine the range of $\|\mathbf{w}\|_2^2$ over Δ_k. For the upper bound, since $w_i \in [0,1]$ and $\sum_i w_i = 1$ for any $\mathbf{w} \in \Delta_k$:

$$\|\mathbf{w}\|_2^2 = \sum_{i=1}^k w_i^2 \le \max_i w_i \cdot \sum_{i=1}^k w_i \le 1.$$

The maximum $\|\mathbf{w}\|_2^2 = 1$ is achieved at the vertices $\mathbf{e}_i$ of the simplex.

For the lower bound, by the Cauchy–Schwarz inequality, we have $\langle \mathbf{1}, \mathbf{w}\rangle^2 \le \|\mathbf{1}\|_2^2 \cdot \|\mathbf{w}\|_2^2$. The LHS equals 1, and the RHS equals $k\|\mathbf{w}\|_2^2$, so we have $1 \le k\|\mathbf{w}\|_2^2$. Therefore:

$$\|\mathbf{w}\|_2^2 \ge 1/k.$$

The minimum $\|\mathbf{w}\|_2^2 = 1/k$ is achieved at the uniform distribution $\mathbf{w} = (1/k, \ldots, 1/k)$. Therefore:

$$R(\mathbf{u}) - \min_{\mathbf{w}\in\Delta_k} R(\mathbf{w}) = \frac{1}{\eta}\left(\|\mathbf{u}\|_2^2 - \min_{\mathbf{w}\in\Delta_k}\|\mathbf{w}\|_2^2\right) = \frac{1}{\eta}\left(\|\mathbf{u}\|_2^2 - \frac{1}{k}\right) \le \frac{1}{\eta}\left(1 - \frac{1}{k}\right) \le \frac{1}{\eta}.$$

□

Corollary E.9. *For dynamic databases under absolute loss, the FTRL algorithm (Algorithm 4) with squared ℓ_2 regularization function $R(\mathbf{w}) = \frac{1}{\eta}\|\mathbf{w}\|_2^2$ and learning rate $\eta = \sqrt{2/T}$ achieves the following regret bound for point queries:*

$$\textit{Regret}_T = \sum_{t=1}^T \textit{Loss}_t(\hat{\mathbf{w}}_t) - \min_{\mathbf{w}\in\Delta_k} \sum_{t=1}^T \textit{Loss}_t(\mathbf{w}) \le \sqrt{2T} = O(\sqrt{T}). \tag{17}$$

Proof. Substituting $\sigma = 2/\eta$ (Lemma E.6), $L = 1$ (Lemma E.7), and the diameter bound (Lemma E.8) at most $\frac{1}{\eta}$ into Theorem E.5: $\text{Regret}_T \le \frac{1}{\eta} + \frac{\eta T}{2}$. Optimizing over η yields $\eta = \sqrt{2/T}$ and $\text{Regret}_T \le \sqrt{2T}$. □

E.2.2 Linear Queries (Theorem 2.2)

For linear queries we instantiate Algorithm 4 with the negative entropy regularization function: $R(\mathbf{w}) = \frac{1}{\eta}\sum_{i=1}^k w_i \log w_i$. The following three lemmas instantiate Theorem E.5, all measured with respect to the ℓ_1 norm.

Lemma E.10. *The negative entropy regularizer $R(\mathbf{w}) = \frac{1}{\eta}\sum_{i=1}^k w_i \log w_i$ is $\frac{1}{\eta}$-strongly convex with respect to the ℓ_1 norm on Δ_k.*

Proof. For a differentiable convex f, f is σ-strongly convex w.r.t. a norm $\|\cdot\|$ iff its Bregman divergence satisfies, for all $\mathbf{p}, \mathbf{q}$,

$$D_f(\mathbf{p}, \mathbf{q}) := f(\mathbf{p}) - f(\mathbf{q}) - \langle \nabla f(\mathbf{q}), \mathbf{p} - \mathbf{q}\rangle \ \ge\ \frac{\sigma}{2}\|\mathbf{p} - \mathbf{q}\|^2.$$

For the negative entropy function $f(\mathbf{w}) = \sum_{i=1}^{k} w_i \log w_i$, we have $\nabla f(\mathbf{w}) = (1 + \log w_i)_{i=1}^{k}$. Thus for any $\mathbf{p}, \mathbf{q} \in \Delta_k$, using $\sum_i p_i = \sum_i q_i = 1$:

$$\begin{aligned} D_f(\mathbf{p}, \mathbf{q}) &= \sum_i p_i \log p_i - \sum_i q_i \log q_i - \sum_i (1 + \log q_i)(p_i - q_i) \\ &= \sum_i p_i \log p_i - \sum_i p_i \log q_i = \sum_i p_i \log \frac{p_i}{q_i} \\ &= \mathrm{KL}(\mathbf{p} \,\|\, \mathbf{q}). \end{aligned}$$

By Pinsker's inequality, for all distributions $\mathbf{p}, \mathbf{q} \in \Delta_k$,

$$\mathrm{KL}(\mathbf{p} \,\|\, \mathbf{q}) \;\geq\; \tfrac{1}{2} \|\mathbf{p} - \mathbf{q}\|_1^2.$$

Therefore $D_f(\mathbf{p}, \mathbf{q}) \geq \frac{1}{2} \|\mathbf{p} - \mathbf{q}\|_1^2$, which is exactly the 1-strong convexity of f w.r.t. $\|\cdot\|_1$.

Finally, strong convexity scales linearly with positive constants. For the function $g = \alpha \cdot f$ for any $\alpha > 0$, if f is σ-strongly convex, then:

$$\begin{aligned} D_g(\mathbf{p}, \mathbf{q}) &= \alpha f(\mathbf{p}) - \alpha f(\mathbf{q}) - \langle \alpha \nabla f(\mathbf{q}), \mathbf{p} - \mathbf{q} \rangle \\ &\geq \alpha \frac{\sigma}{2} \|\mathbf{p} - \mathbf{q}\|^2. \end{aligned}$$

Thus g is $\alpha\sigma$-strongly convex. Applying this to our case, we get that $R_\eta = \frac{1}{\eta} \cdot \sum_{i=1}^{k} w_i \log w_i$ is $\frac{1}{\eta}$-strongly convex w.r.t. $\|\cdot\|_1$. □

Lemma E.11. *For any $\sigma_t \in [0, 1]$ and $\mathbf{v}_t \in [0, 1]^k$, the loss function $\mathit{Loss}_t(\mathbf{w}) = |\sigma_t - \langle \mathbf{v}_t, \mathbf{w} \rangle|$ is 1-Lipschitz with respect to the ℓ_1 norm on Δ_k.*

Proof. For any $\mathbf{w}, \mathbf{w}' \in \Delta_k$:

$$\begin{aligned} |\mathrm{Loss}_t(\mathbf{w}) - \mathrm{Loss}_t(\mathbf{w}')| &= ||\sigma_t - \langle \mathbf{v}_t, \mathbf{w} \rangle| - |\sigma_t - \langle \mathbf{v}_t, \mathbf{w}' \rangle|| \\ &\leq |\langle \mathbf{v}_t, \mathbf{w} \rangle - \langle \mathbf{v}_t, \mathbf{w}' \rangle| \quad \text{(triangle inequality)} \\ &= |\langle \mathbf{v}_t, \mathbf{w} - \mathbf{w}' \rangle| \\ &\leq \|\mathbf{v}_t\|_\infty \cdot \|\mathbf{w} - \mathbf{w}'\|_1 \quad \text{(Hölder's inequality)} \\ &\leq \|\mathbf{w} - \mathbf{w}'\|_1, \end{aligned}$$

where the last inequality uses $\|\mathbf{v}_t\|_\infty \leq 1$ since $\mathbf{v}_t \in [0, 1]^k$. □

Lemma E.12. *For the negative entropy regularizer $R(\mathbf{w}) = \frac{1}{\eta} \sum_{i=1}^{k} w_i \log w_i$ on Δ_k, and any $\mathbf{u} \in \Delta_k$: $R(\mathbf{u}) - \min_{\mathbf{w} \in \Delta_k} R(\mathbf{w}) \leq \frac{\log k}{\eta}$.*

Proof. The negative entropy $-\sum_{i=1}^{k} w_i \log w_i$ is maximized on Δ_k by the uniform distribution $\mathbf{w} = (1/k, \ldots, 1/k)$, which gives entropy $\log k$. The minimum is 0, achieved when the distribution is concentrated on a single element. Therefore:

$$R(\mathbf{u}) - \min_{\mathbf{w} \in \Delta_k} R(\mathbf{w}) = \frac{1}{\eta} \left[\sum_{i=1}^{k} u_i \log u_i - \min_{\mathbf{w} \in \Delta_k} \sum_{i=1}^{k} w_i \log w_i \right] \leq \frac{1}{\eta} (0 - (-\log k)) = \frac{\log k}{\eta}. \quad (18)$$

□

With these lemmas in place, substituting $\sigma = 1/\eta$, $L = 1$ and applying the diameter bound from Lemma E.12 into Theorem E.5, we derive the main upper bound:

Corollary E.13. *For dynamic databases under absolute loss, the FTRL algorithm (Algorithm 4) with negative entropy regularization function $R(\mathbf{w}) = \frac{1}{\eta} \sum_{i=1}^{k} w_i \log w_i$ and learning rate $\eta = \sqrt{\frac{\log k}{T}}$ achieves the following regret bound for linear queries:*

$$\mathit{Regret}_T = \sum_{t=1}^{T} \mathit{Loss}_t(\widehat{\mathbf{w}}_t) - \min_{\mathbf{w} \in \Delta_k} \sum_{t=1}^{T} \mathit{Loss}_t(\mathbf{w}) \leq 2\sqrt{T \log k} = O(\sqrt{T \log k}) \quad (19)$$

F Experimental Details and Additional Results

In this section, we provide the experimental details and additional results that complement Section 6. Section F.1 specifies the databases, query workloads, algorithms, and baselines. Section F.2 reports the full comparison with the baselines, including the frozen baselines and the Power and Forest databases. Section F.3 reports regret for range queries and on Power and Forest, and compares it with the theoretical bounds. Section F.4 reports the runtime. Unless stated otherwise, we set $k = 200$ and $T = 500$, and every reported metric is a mean over five query streams; the databases are the same for all seeds.

F.1 Experimental Setup

Datasets. Each database is the normalized histogram of one numerical attribute over k bins of fixed width. Taxi uses the `trip_distance` attribute of the 24 monthly NYC yellow-taxi trip files of 2019–2020 (NYC Taxi and Limousine Commission, 2024) (108.2M tuples); Power uses the `Global_active_power` attribute of the household electric power consumption data (Dua et al., 2017) (2.0M tuples); Forest uses the `Elevation` attribute of the forest cover type data (Dua et al., 2017) (581K tuples). The bin widths are 0.1 mile, 0.02 kW, and 10 m, respectively. The database of support size k consists of the tuples in the first k bins, and $\mathbf{w}^* \in \Delta_k$ is their normalized histogram. Increasing k therefore extends the domain rather than refining the bins. We evaluate k from 20 to 200. Figure 2 shows the three histograms and the nested domains.

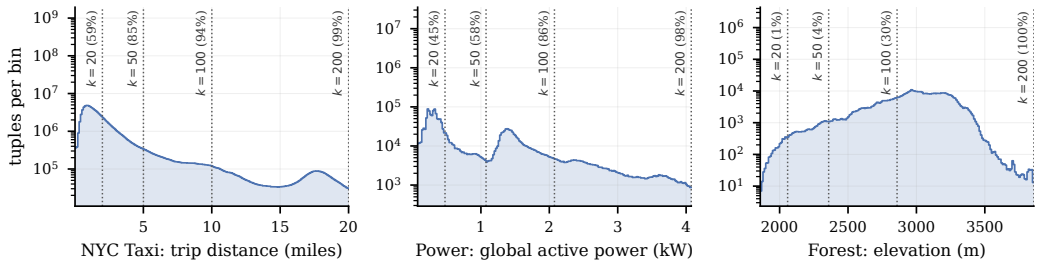

Figure 2: Static histograms of the three databases over the 200 fixed-width bins (log scale). Dotted lines mark the right end of the domain for support size $k \in \{20, 50, 100, 200\}$, labeled with the fraction of all tuples that the database of that support size contains.

Static and dynamic databases. The static database of each dataset is the histogram of all its tuples. The dynamic database is defined for Taxi only: the database at round t is the histogram, over the same bins, of the trips picked up on day t, where day 1 is January 1, 2019. The database therefore changes in every round, and $T = 500$ ends on May 14, 2020. Figure 3 shows how the dynamic database changes. From mid-March 2020 (round 441), the number of trips falls drastically, and the database distribution changes substantially from one day to the next.

Query workloads. At round t, the query is sampled from a window with center c_t and width B_t on the domain rescaled to $[0, 1]$, and the workload determines how this window moves. Under the abrupt workload, the T rounds form five phases of equal length; c_t is constant within a phase and jumps to a new random center, at least 0.25 away, at each phase change, and $B_t = 0.5$. Under the oscillating workload, $c_t = 0.5 - 0.4\cos(2\pi(t-1)/80)$ sweeps between 0.1 and 0.9 with a period of 80 rounds, and $B_t \sim \mathrm{Uniform}[0.3, 0.5]$.

Given the window, a range query $\mathbf{v}_t$ is the indicator vector of the bins covered by an interval of length $h_t \sim \mathrm{Uniform}[0.1, 0.3]$ at a random position inside the window; its endpoints are clipped to $[0, 1]$ and rounded outward to bin boundaries. A linear query sets $v_{t,i} \sim \mathrm{Uniform}[0, 1]$ independently for every bin i whose center lies in the window, and $v_{t,i} = 0$ otherwise. For a given seed, all methods and both database settings receive the same query stream.

Algorithm implementation details. SeqMaxEnt computes the maximum-entropy histogram numerically by solving its convex dual with L-BFGS-B (Byrd et al., 1995); over all static runs, the

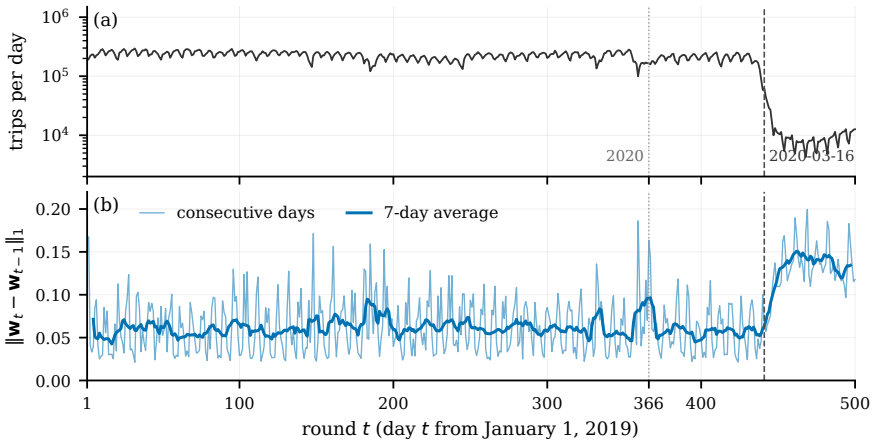

Figure 3: The dynamic Taxi database over the $T = 500$ rounds (round t is day t from January 1, 2019). (a) Trips per day. (b) ℓ_1 distance between the database vectors of consecutive days at $k = 200$, with its 7-day moving average. The dotted line marks January 1, 2020, and the dashed line March 16, 2020 (round 441).

Table 4: Full version of Table 1 on the Taxi databases (range queries, $k = 200$, $T = 500$, scored rounds 41–500, mean over five query streams). Baselines are either frozen after their initial fit on rounds 1–40 (-F) or refitted from scratch on all observed labels every 20 rounds (-R). The lowest value in each column is in bold.

	Static database						Dynamic database					
Queries	abrupt			oscillating			abrupt			oscillating		
Method	sq. loss	abs. loss	q-err$_{90}$	sq. loss	abs. loss	q-err$_{90}$	sq. loss	abs. loss	q-err$_{90}$	sq. loss	abs. loss	q-err$_{90}$
SeqMaxEnt	**0.133**	**0.893**	**1.006**	**0.0009**	**0.099**	**1.003**	–	–	–	–	–	–
ONS	0.321	4.71	1.341	0.120	3.95	1.246	**0.455**	**7.55**	**1.653**	0.225	6.36	1.550
FTRL	1.43	10.4	2.288	0.338	5.59	1.350	1.44	11.8	2.236	0.375	7.22	1.527
NeuroCDF-F	32.9	57.1	26.3	0.059	3.88	2.353	32.7	59.2	22.4	0.216	6.76	2.670
NeuroCDF-R	2.45	7.59	8.128	0.023	2.41	1.593	2.61	10.2	3.918	**0.144**	**5.39**	1.784
MSCN-F	32.5	60.4	925	0.919	9.95	5.004	33.5	62.9	1010	0.616	10.8	882
MSCN-R	2.77	10.6	68.6	0.076	3.60	1.794	2.84	12.5	359	0.241	6.53	3.290
PtsHist-F	52.0	85.0	2.7e7	0.387	6.27	1.320	51.4	85.6	5.4e4	0.584	9.25	1.589
PtsHist-R	5.29	10.8	1.183	0.119	1.83	1.093	5.33	14.4	1.829	0.263	5.84	**1.390**

numerical error, i.e., the largest deviation of a computed histogram from an observed selectivity, is at most 7.3×10^{-10}. ONS starts from the uniform histogram. For $k \leq 64$, we compute its generalized projection onto Δ_k (Hazan, 2016) with SLSQP (Kraft, 1988); for $k > 64$, we use the Euclidean projection (Duchi et al., 2008) instead to save computation, which closely approximates the generalized projection in our runs. FTRL is implemented on the linearized absolute losses, i.e., as exponential weights on their subgradients, which retains the bound of Corollary E.13.

Baselines. Every baseline is first fit on the labels of rounds 1–40. The frozen variant (-F) keeps this model, and the retrained variant (-R) fits a new model from scratch on all labels observed so far every 20 rounds. All methods are scored on rounds 41–500.

F.2 Comparison with Baselines

Table 4 extends Table 1 with the frozen baselines, and Table 5 repeats the static comparison on Power and Forest.

Freezing the baselines after their initial fit is costly, especially under the abrupt workload (Tables 4 and 5): the first 40 queries fall in the region of the first phase, so the frozen models are later queried on regions they were never trained on, and their squared loss is 8 to 17 times that of the retrained ones. Under the oscillating workload, the window already sweeps across the whole domain in

Table 5: Comparison with the baselines on the static Power and Forest databases, with the protocol of Table 4 (range queries, $k = 200$, $T = 500$, scored rounds 41–500, mean over five query streams). The lowest value in each column is in bold.

	Power (static)						Forest (static)					
Queries	abrupt			oscillating			abrupt			oscillating		
Method	sq. loss	abs. loss	q-err$_{90}$	sq. loss	abs. loss	q-err$_{90}$	sq. loss	abs. loss	q-err$_{90}$	sq. loss	abs. loss	q-err$_{90}$
SEQMAXENT	**0.063**	**0.761**	**1.005**	**0.0021**	**0.162**	**1.004**	**0.036**	**0.473**	**1.005**	**0.0001**	**0.055**	**1.003**
ONS	0.228	4.97	1.271	0.142	4.72	1.197	0.119	3.28	1.411	0.035	2.49	1.268
FTRL	0.638	8.12	1.449	0.245	5.47	1.211	0.339	6.08	2.105	0.133	4.17	1.299
NEUROCDF-F	17.8	45.4	12.4	0.052	3.70	1.405	2.59	16.7	9.418	0.011	1.72	1.405
NEUROCDF-R	1.67	7.54	1.746	0.022	2.49	1.249	0.316	3.22	1.826	0.0065	1.34	1.316
MSCN-F	22.8	66.6	17.1	0.578	10.1	1.572	30.2	74.7	58.3	0.132	4.68	1.816
MSCN-R	2.53	13.2	1.888	0.295	7.35	1.415	3.18	11.2	4.059	0.029	2.40	1.670
PTSHIST-F	33.2	74.9	3.9e5	0.346	6.83	1.301	24.8	60.3	1.0e5	0.250	6.72	1.421
PTSHIST-R	3.63	10.1	1.186	0.078	1.98	1.086	1.48	6.06	1.377	0.032	1.69	1.121

the first 40 rounds, and the gap is smaller for most baselines. The extreme q-errors of the frozen PTSHIST arise because it predicts zero for queries that contain none of its points.

On Power and Forest (Table 5), the results mirror those on the static Taxi database in Table 1: SEQMAXENT attains the lowest value of every metric under both workloads, with 9 to 83 times lower squared loss than the best baseline and a 90th-percentile q-error of at most 1.005.

Table 1 also evaluates ONS and FTRL under both losses: ONS attains the lower loss under both, while FTRL is the one with a worst-case guarantee under absolute loss (Corollary E.13).

F.3 ADDITIONAL REGRET RESULTS

Range queries on Taxi. We report the regret for range queries on the static and dynamic Taxi databases in Figure 4. On the static database, the squared regret of SEQMAXENT grows from 0.015 at $k = 20$ to 0.214 at $k = 200$ under the abrupt workload and from 0.021 to 0.375 under the oscillating workload; its absolute regret at $k = 200$ is 1.48 and 1.14, respectively. On the dynamic database, the squared regret of ONS grows from 0.039 to 0.557 (abrupt) and from 0.031 to 0.488 (oscillating), and the absolute regret of FTRL from 2.66 to 11.08 and from 2.50 to 11.17. Along the stream, the static regret flattens as more queries become linear combinations of earlier ones, whose answers SEQMAXENT predicts exactly, and it rises in steps after the phase changes of the abrupt workload, when queries move to regions with few constraints. The dynamic regret peaks between rounds 420 and 460 and then decreases slightly: after mid-March 2020, the loss of the best fixed histogram grows faster than that of the learners (Section F.1).

Power and Forest. We report the regret of SEQMAXENT on the static Power and Forest databases for both query classes in Figure 5. The regret need not increase with k: it follows how far the database is from the uniform histogram $\mathbf{u}$, measured by $2D_{\mathrm{KL}}(\mathbf{w}^*\|\mathbf{u})$, which is not monotone in k on these databases because extending the domain changes the shape of the renormalized histogram. On Power, for instance, both drop from $k = 50$ to $k = 80$, when the domain grows to include the second mode of the distribution. Along the stream, the regret flattens early: at least 96% of the final regret is incurred in the first 250 rounds.

Comparison with the regret bounds. Besides the worst-case bound $2\ln k$ of Proposition 3.1, its proof yields the instance-dependent bound $2D_{\mathrm{KL}}(\mathbf{w}^*\|\mathbf{u})$, where $\mathbf{u}$ is the uniform histogram: the telescoping sum of Lemmas 3.2 and 3.3 ends at $2D_{\mathrm{KL}}(\hat{\mathbf{w}}_{T+1}\|\mathbf{u})$, and $\hat{\mathbf{w}}_{T+1}$ minimizes $D_{\mathrm{KL}}(\cdot\|\mathbf{u})$ over a set of histograms that contains $\mathbf{w}^*$. At $k = 200$, the squared regret of SEQMAXENT on the three static databases is at most 0.375, far below both $2\ln k = 10.6$ and $2D_{\mathrm{KL}}(\mathbf{w}^*\|\mathbf{u})$, which ranges from 1.24 to 2.26; its absolute regret is at most 1.72, far below $\sqrt{2(k-1)\ln k} = 45.9$ (Corollary C.4).

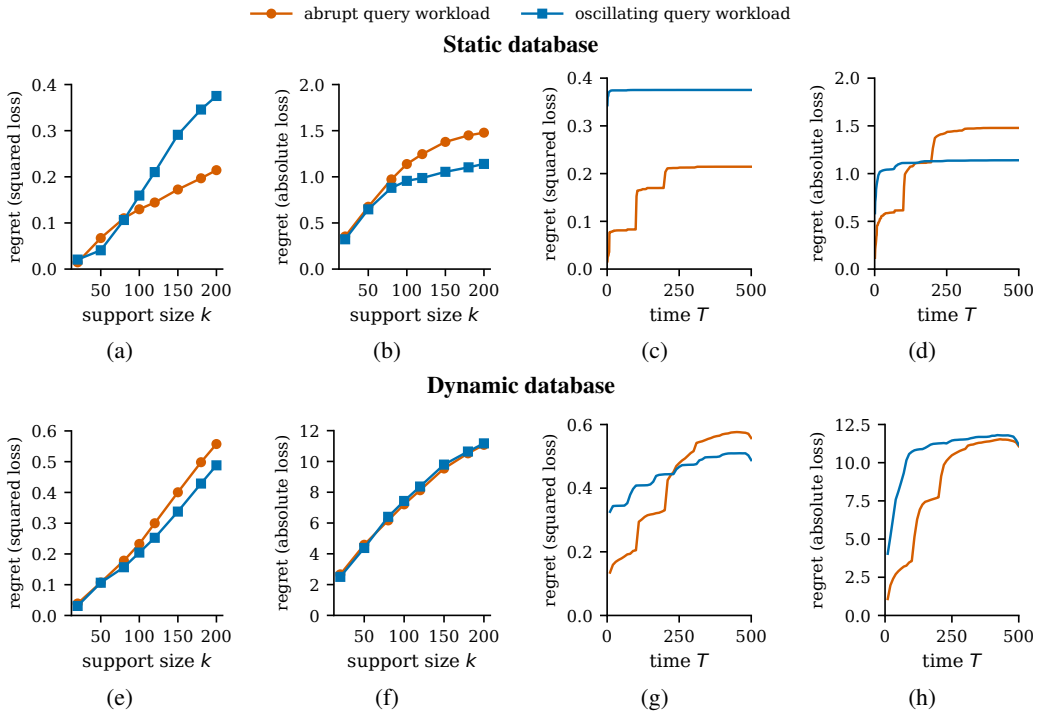

Figure 4: Regret for range queries on the static (top; SEQMAXENT) and dynamic (bottom; ONS under squared loss, FTRL under absolute loss) Taxi databases, as k varies with $T = 500$ (a, b, e, f) and along 500-round streams with $k = 200$ (c, d, g, h), under squared loss (a, c, e, g) and absolute loss (b, d, f, h).

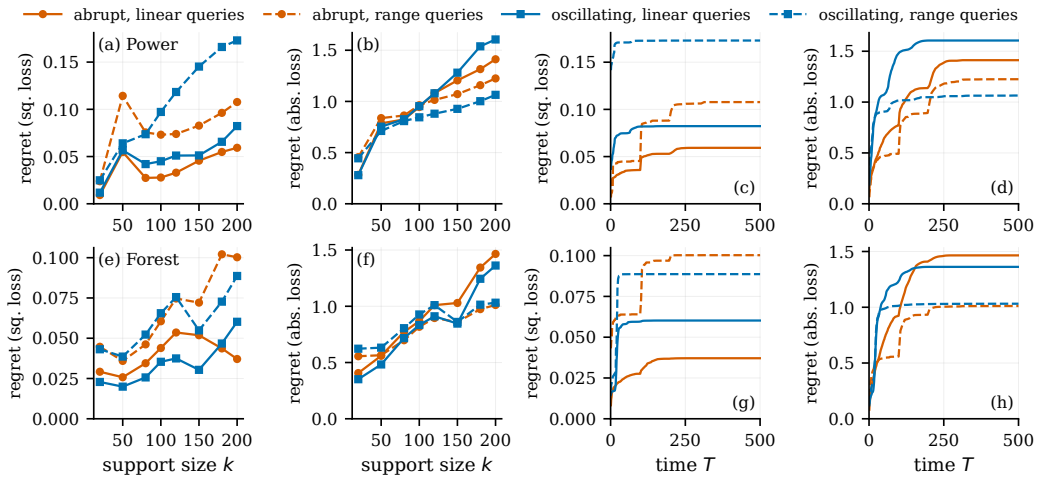

Figure 5: Regret of SEQMAXENT on the static Power (top) and Forest (bottom) databases for linear (solid) and range (dashed) queries, as k varies with $T = 500$ (a, b, e, f) and along 500-round streams with $k = 200$ (c, d, g, h), under squared loss (a, c, e, g) and absolute loss (b, d, f, h).

F.4 RUNTIME

Table 6 reports the wall-clock runtime on an Intel Core i7-12700K with one CPU thread per process. For our learners, the totals include prediction and update in all 500 rounds; for the baselines, the training totals include the initial fit and the 22 refits, and prediction is reported separately. The measurements exclude data preparation and the offline computation of the comparators. At $k =$

Table 6: Runtime on the Taxi databases (range queries, $k = 200$, $T = 500$; mean over five query streams and both workloads; one CPU thread per process). Times are in seconds unless marked.

Database	Method	Initial fit	Per round or refit	Total per stream	Prediction (ms/query)
static	SEQMAXENT	–	2.02 ms	1.01	–
static	NEUROCDF	0.88	0.80	18.51	0.058
static	MSCN	0.53	0.73	16.49	0.081
static	PTSHIST	0.11	0.10	2.39	0.141
dynamic	ONS	–	0.135 ms	0.068	–
dynamic	FTRL	–	0.016 ms	0.008	–
dynamic	NEUROCDF	0.87	0.84	19.33	0.064
dynamic	MSCN	0.89	0.79	18.34	0.088
dynamic	PTSHIST	0.18	0.10	2.40	0.137

200, a stream takes about 1.0 second for SEQMAXENT, 0.07 seconds for ONS with the Euclidean projection, and 0.01 seconds for FTRL, whereas training a baseline takes 2.4 to 19.3 seconds per stream.