ISYE 8813: Algorithmic Foundations of Ethical ML **Fall 2021**

Lectures 11-12: Online MW

Lecturer: Juba Ziani

SmallDB required us to know all our queries in advance. We will now see a mechanism, Private Multiplicative Weights (PMW), which allows queries to be chosen adaptively. This mechanism will be a combination of Sparse Vector (to answer threshold queries adaptively) and the multiplicative weight algorithm for learning linear predictors online.

1 High-level Summary of Private MW

View database $x \in \mathbb{N}^{|\mathcal{X}|}$ as histogram, and consider only linear queries (i.e., linear functions of histograms). Then answering linear queries is the same problem as learning the linear function x that defines query answers $\langle x, q \rangle$, given a query $q \in [0,1]^{|\mathcal{X}|}$. This is the same thing we did for SmallDB: we said, "I'm not going to give you answers to your queries. Instead I'll give you a database and you can answer your own queries." However, SmallDB comes with the caveat that we need to know the class of queries Q that we want to answer in advance; the choice of small database that we pick to answer our queries on crucially depends on it.

Then, what happens in the online setting, when I need to answer queries now before I see what queries arrive next? What if I am doing some adaptive data analysis and I want to pick the next query as a function of the answers to the previous query? One way to do so is to use the Multiplicative Weight Algorithm (for linear queries), and to make it differentially private.

Private Multiplicative weights can learn any linear predictor by making only a small number of queries – which is good for our privacy budget. It maintains a public "hypothesis predictor" and accesses the data only by requiring examples of queries where the hypothesis differs greatly from the true data... which at this point should remind you that we saw a way to do this just two weeks ago: SparseVector!

In this lecture, we will start with regular (non-private) MW, and then later we'll make it private by using SparseVector.

2 Regular (non-private) MW

We will think of a database x as being a probability distribution over data universe $\mathcal{X}$. That is, let $\Delta([\mathcal{X}])$ denote the set of all distributions over the set $[|\mathcal{X}|]$, and we have $x \in \Delta([\mathcal{X}])$. This corresponds to scaling down every entry in the histogram version of x by a factor $\frac{1}{\|x\|_1}$ to ensure the entries sum to 1. Then the i-th entry of x corresponds to the fraction of data points that are of type $\mathcal{X}_i$. We will assume that there is a true database x that we do not

have access to and do not know, that determines the true answer to the queries $f_1, \ldots, f_k$ that we run.

The set-up is then going to be the following:

- Queries will arrive sequentially. At each time step t, we receive a single query f_t. We observe an estimate of the answer to that query $v_t \sim f_t(x)$ (but note that we still do not see the true database x). Note here that we allow $v_t \neq f_t(x)$ (but we will expect v_t to still be close to $f_t(x)$): this will be important when moving to the private version of the algorithm, because we will want v_t to be a differentially private estimate of $f_t(x)$.
- Our goal will then be to find a sequence of distributions $x^t = x_1^t, \ldots, x_{|\mathcal{X}|}^t$ on the data universe $\mathcal{X}$ (think of x_i' as the weight on $\mathcal{X}_i$, or the probability of an element in our database being $\mathcal{X}_i$), one such distribution/database for each time step, to guarantee accuracy. x^{t+1} is picked as a function of the history up until time t.
- Note that due to the online learning nature of the problem, we will necessarily be making mistakes in the beginning of the algorithm, as long as we have not yet seen enough data to learn how to answer our queries correctly. Hence, our accuracy objective is going to be to limit the number of time steps in which $|f_t(x^t) - f_t(x)|$ is large.

MW is instantiated with a learning rate parameter $\eta \leq 1$. We leave this as a free parameter for now, but in later analysis, we will set $\eta = \alpha/2$. This rule will be applied iteratively to the public hypothesis predictor x^t, so we will index all inputs and outputs by t, corresponding to the t-th update step. The algorithm takes in the current public hypothesis x^t, a query f_t on which the public hypothesis performs poorly (i.e., $f_t(x_t)$ is far from $f_t(x)$), and v_t, which is an estimate of query's answer on the true database: $f_t(x)$. After a single update step, it produces the next public hypothesis x^{t+1}, which should provide a more correct answer to query f_t.

Algorithm 1: MW Update Rule: MW(x^t, f_t, v_t)

Input: public hypothesis database x^t, query f_t on which the public hypothesis performs poorly, estimate v_t of true answer $f_t(x)$ on true database x

if $v_t < f_t(x^t)$ **then**
  Let $r_t = f_t$
else
  Let $r_t = 1 - f_t$
end

Update: for all $i \in |\mathcal{X}|$, let:

$\hat{x}_i^{t+1} = \exp(-\eta r_t[i]) x_i^t$

$x_i^{t+1} = \frac{\hat{x}_i^{t+1}}{\sum_{j=1}^{|\mathcal{X}|} \hat{x}_j^{t+1}}$

Output: x^{t+1}.

The update step is the most straightforward: the algorithm updates each entry of the database x^t by a multiplicative factor (hence the name!) that depends exponentially on

the algorithm's step size η and the direction in which the hypothesis was wrong, relative to the true database. These multiplicative updates may not result in a valid probability distribution, so the entries of the database are renormalized to ensure they sum to 1.

Now let's revisit the sign flipping step. We're creating a new vector that corresponds to the function f_t, and adjusting this vector to account for the direction in which the hypothesis was wrong. Intuitively, if $f_t(x) < f_t(x^t)$, then the query answer on the hypothesis database is too large, and we need to down-weight the entries of x^t that cause large values of f_t, and vice versa in the alternative case. So, in the case where $f_t(x^t)$ is too large, we want to give more weight to the elements $\mathcal{X}_i$ in the universe where $f_t(\mathcal{X}_i)$ is small, and less weight to the elements where it is big. So, in that case, by letting $r_t = f_t$, we have that more weight is given to the elements with the smaller r_t, hence the smaller values of f_t, as we wanted!

3 MW Convergence

The main result that we will show for non-private MW is that (under certain technical conditions), it converges with a small number of updates.

Theorem 1. *Fix a class of linear queries Q and a database $x \in \Delta([|\mathcal{X}|])$, and let $x^1 \in \Delta([|\mathcal{X}|])$ be the uniform distribution over $\mathcal{X}$. That is, let $x^1_i = \frac{1}{|\mathcal{X}|}$ for all i. Now consider a maximal length sequence of databases x^t for $t \in \{2, \ldots, L\}$ generated by setting $x^{t+1} = MW(x^t, f_t, v_t)$ with learning rate $\eta = \alpha/2$, where for each t, $f_t \in Q$ and $v_t \in \mathbb{R}$,*

1. $|f_t(x) - f_t(x^t)| > \alpha$ *and*
2. $|f_t(x) - v_t| < \alpha$.

Then $L \leq \frac{4 \log |\mathcal{X}|}{\alpha^2}$.

Note that on the last database x^{L+1}, it must be that for all $f \in Q$, $|f(x) - f(x^{L+1})| \leq \alpha$, otherwise we could extend the sequence, contradicting maximality. This theorem says if you re-run MW many times, always fining some query f_t on which you are doing poorly (i.e., $f_t(x^t)$ is far from $f_t(x)$), then after L rounds, you will have to stop because you are doing well on all queries.

Private MW (which we will see next lecture) works by maintaining a public approximation x^t to the database x. Given an input query f, the mechanism will check the (noisy) difference $|f(x) - f(x^t)|$ to see if the public approximation x^t is good or bad with respect to this query. If the noisy difference is large, then Private MW calls the Laplace Mechanism to produce a noisy approximation $v^t (= f(x) + Lap)$ to the true answer $f(x)$, and then the MW update rule called with parameters (x^t, f, v_t). This is where SparseVector comes in: we only need to update our noisy approximation when it does poorly, i.e. when $|f(x) - f(x^t)|$ is large enough/bigger than some threshold; the rest of the time, we can decide to output nothing.

If this update rule is invoked only when x_t is truly bad on f (i.e., when $f(x) - f(x^t)$ large, Condition 1 of Theorem 8), and if the approximation v_t is sufficiently accurate (Condition 2 of Theorem 8), then Theorem 8 tells us that we do not need to do too many updates

(because L is bounded) and the resulting x^{L+1} gives accurate answers to all queries in Q (because we can not find another bad query).

We will write this formally as algorithm in the next lecture, but we will first prove this theorem. We will do it by keeping track if a potential function Ψ to measure the similarity between the public hypothesis x_t at time t, and the true database x. We will show three key facts:

1. Ψ does not start out too large.
2. Ψ decreases significantly with each update
3. Ψ is always non-negative.

These three things will show that we do not need too many update rounds.

Quick note: This proof technique, called a "potential argument", is used often in analyzing algorithms. These same three steps are used in all proofs of this style. The critical step of the analysis is choosing a potential function that captures progress of the algorithm toward its final goal.

Proof of Theorem 8. We define our potential function to be the relative entropy or KL-divergence between x and x^t:

$$\Psi_t \triangleq KL(x||x^t) = \sum_{i=1}^{|\mathcal{X}|} x_i \log\left(\frac{x_i}{x_i^t}\right).$$

Proposition 2 (Condition 3)**.** *For all t, $\Psi_t \geq 0$.*

Proof. Relative entropy is always non-negative by the log-sum inequality, while states that if $a_1, \ldots, a_n$ and $b_1, \ldots, b_n$ are all non-negative and sum to 1, then

$$\Psi = \sum_{i=1}^{n} a_i \log \frac{a_i}{b_i} \geq \sum_i a_i \log \frac{\sum_i a_i}{\sum_i b_i} = 0.$$

□

Proposition 3 (Condition 1)**.** $\Psi_1 \leq \log|\mathcal{X}|$.

Proof. Recall that $x_i^1 = \frac{1}{|\mathcal{X}|}$ for all i, so

$$\Psi_1 = \sum_{i=1}^{|\mathcal{X}|} x_i \log(|\mathcal{X}|x_i).$$

This quantity is maximized when $\exists j$ such that $x[j] = 1$, and $x_i = 0$ for all $i \neq j$, giving $\Psi_1 = \log|\mathcal{X}|$ □

We have done the two easy parts. Now we need to show that Ψ drops by at least $\frac{\alpha^2}{4}$ with every update. Because Ψ starts at $\log|\mathcal{X}|$ and can not go negative, then there can be at most $L \leq \frac{4\log|\mathcal{X}|}{\alpha^2}$ update steps.

Lemma 4 (Condition 2)). [1] *For all* $\Psi_t - \Psi_{t+1} \geq \eta(\langle r_t, x^t\rangle - \langle r_t, x\rangle) - \eta^2$

Proof.

$$
\begin{aligned}
\Psi_t - \Psi_{t+1} &= \sum_{i=1}^{|\mathcal{X}|} x_i \log\left(\frac{x_i}{x_i^t}\right) - \sum_{i=1}^{|\mathcal{X}|} x_i \log\left(\frac{x_i}{x_i^{t+1}}\right) \\
&= \sum_{i=1}^{|\mathcal{X}|} x_i \left(\log\left(\frac{x_i}{x_i^t}\right) - \log\left(\frac{x_i}{x_i^{t+1}}\right)\right) && \text{(dist. prop.)} \\
&= \sum_{i=1}^{|\mathcal{X}|} x_i \log\left(\frac{x_i^{t+1}}{x_i^t}\right) && \text{(log rule)} \\
&= \sum_{i=1}^{|\mathcal{X}|} x_i \log\left(\frac{x_i^{t+1}/\sum_{i=1}^{|\mathcal{X}|}\hat{x}_i^{t+1}}{x_i^t}\right) && \text{(def. of } x_i^{t+1}\text{)} \\
&= \sum_{i=1}^{|\mathcal{X}|} x_i \left[\log\left(\frac{x_i^t.\exp(-\eta r_t[i])}{x_i^t}\right) - \log\left(\sum_{j=1}^{|\mathcal{X}|}\exp(-\eta r_t[j]x_j^t)\right)\right] && \text{(log rule)} \\
&= -\left(\sum_{i=1}^{|\mathcal{X}|} x_i \eta r_t[i]\right) - \log\left(\sum_{j=1}^{|\mathcal{X}|}\exp(-\eta r_t[j])x_j^t\right) && \text{(cancel first term, dist sum, } \textstyle\sum_i \\
&\overset{(*)}{=} -\eta\langle r_t, x\rangle - \log\left(\sum_{j=1}^{|\mathcal{X}|}\exp(-\eta r_t[j])x_j^t\right) && \text{(def. of } \langle\cdot,\cdot\rangle\text{, keep second term}
\end{aligned}
$$

Before proceeding, let's look at this second term.

$$\exp(-\eta r_t[j]) \leq 1 - \eta r_t[j] + \frac{\eta^2 (r_t[j])^2}{2} \leq 1 - \eta r_t[j] + \eta^2,$$

where the first inequality comes from a Taylor expansion and the second inequality comes from the fact that $\eta r_t[j] \leq 1$.

[1] Later we will set $\eta = \frac{\alpha}{2}$

Using this to continue to bound (*):

$$\begin{aligned}
(*) &\geq -\eta\langle r_t, x\rangle - \log\left(\sum_{j=1}^{|\mathcal{X}|} x_j^t(1+\eta^2-\eta r_t[j])\right) \\
&= -\eta\langle r_t, x\rangle - \log\sum_{j=1}^{|\mathcal{X}|}(1+\eta^2-\eta\sum_j x_j^t r_t[j]) && \text{(pull sum inside)} \\
&= \eta\langle r_t, x\rangle - (\eta^2 - \eta\langle r_t, x^t\rangle) && (\log(1+y)\leq y \text{ for } y>-1) \\
&= \eta(\langle r_t, x^t\rangle - \langle r_t, x\rangle) - \eta^2 && \text{(collect terms)}
\end{aligned}$$

□

Now let's finish the proof of Theorem 8. We have $|f_t(x) - f_t(x^t)| \geq \alpha$ and $|v_t - f_t(x)| < \alpha$ (by assumption), so $f_t(x) < f_t(x^t)$ iff $v_t < f_t(x^t)$. Also from the algorithm, $r_t = f_t$ if $f_t(x^t) - f_t(x) \geq \alpha$ and $r_t = 1 - f_t$ if $-f_t(x^t) + f_t(x) \geq \alpha$. Therefore, $(\langle r_t, x^t\rangle - \langle r_t, x\rangle) \geq \alpha$. By Lemma 4 and setting $\eta = \frac{\alpha}{2}$,

$$\Psi_t - \Psi_{t+1} \geq \frac{\alpha}{2}(\alpha) - \frac{\alpha^2}{4} = \frac{\alpha^2}{2} - \frac{\alpha^2}{4} = \frac{\alpha^2}{4}.$$

Finally,

$$0 \leq \Psi_L \leq \Psi_0 - L\frac{\alpha^2}{4} \leq \log(|\mathcal{X}|) - L\frac{\alpha^2}{4} \quad \Rightarrow \quad L \leq \frac{4\log(|\mathcal{X}|)}{\alpha^2}.$$

□

4 Building up to Private MW

The queries to SparseVector will be about the error $|f_i(x) - f_i(x^t)|$, and we will call these answers E_i to emphasize that we're asking about the *error*, not about the function's value. SparseVector can only tell us if something is above or below a threshold. We'll capture the absolute value by running two copies of SparseVector to ask about both $f_i(x) - f_i(x^t)$ and $f_i(x^t) - f_i(x)$.

Theorem 5 (PMW privacy)**.** *PrivateMW is (ε,δ)-DP*

The proof of Theorem 5 follows immediately from the privacy of SparseVector because PrivateMW only accesses the data through SparseVector. Everything else is post-processing.

Theorem 6 (PMW accuracy)**.** *With probability* $(1-\beta)$*, for all* f_i*, PrivateMW returns an answer* a_i *such that* $|f_i(x) - a_i| \leq 3\alpha$ *for:*

$$\alpha = \left(\frac{36\log|\mathcal{X}|\Big(\log 2|Q| + \log(\frac{32\log|\mathcal{X}|^{1/3}\|x\|_1^{2/3}}{\beta})\Big)}{\|x\|_1^2\varepsilon}\right)^{\frac{1}{3}} \qquad \text{if } \delta = 0, \text{ and}$$

Algorithm 2: PrivateMW($x, \{f_i\}, \epsilon, \delta, \alpha, \beta, Q$)

Let $c = \frac{4 \log |\mathcal{X}|}{\alpha^2}$;
if $\delta = 0$ **then**
  Let $T = \dfrac{18c(\log(2|Q|) + \log(\frac{4c}{\beta}))}{\varepsilon \|x\|_1}$;
else
  Let $T = \dfrac{(2 + 32\sqrt{2})\sqrt{c \log(2/\delta)}(\log(k) + \log(\frac{4c}{\beta}))}{\varepsilon \|x\|_1}$;
end
Initialize Sparse($x, \{f_i\}, T, c, \varepsilon, \delta$), outputting $\{E_i\}$;
Let $t = 0$, $x^0 \in \Delta([|\mathcal{X}|])$ s.t. $x_i^0 = \frac{1}{|\mathcal{X}|}$ $\forall i \in [|\mathcal{X}|]$;
for *each query* f_i **do**
  Let $f'_{2i-1}(x) = f_i(x) - f_i(x^t)$;
  Let $f'_{2i}(x) = f_i(x^t) - f_i(x)$;
  if $E_{2i-1} = \perp$ *and* $E_{2i} = \perp$ **then**
    Let $a_i = f_i(x^t)$;
  else
    if $E_{2i-1} \in \mathbb{R}$ **then**
      Let $a_i = f_i(x^t) + E_{2i-1}$;
    else
      Let $a_i = f_i(x^t) - E_{2i}$;
    end
    Let $x^{t+1} = MW(x^t, f_i, a_i)$;
    Let $t = t + 1$;
  end
end

$$\alpha = \left(\frac{(2 + 32\sqrt{2})\sqrt{\log |\mathcal{X}| \log \frac{2}{\delta}}(\log 2|Q| + \log \frac{32\|x\|_1}{\beta})}{\|x\|_1 \varepsilon} \right)^{\frac{1}{2}} \qquad \text{if } \delta > 0.$$

Note that when $\delta > 0$, we get better accuracy in terms of $\|x\|_1$ because of the better composition theorems for (ε, δ)-DP.

Proof sketch. Use SparseVector accuracy theorem to show that w.h.p,

1. MW update rule is only called when $|f_i(x) - f_i(x^t)|$ is large. In this case, we have a noisy, private answer to $f_i(x) - f_i(x^t)$ that we can use to correct for the error on $f_i(x^t)$ and output an accurate a_i.

2. The released noisy approximation to $f_i(x)$ is accurate the rest of the time, because we only call the MW update rule when $|f_i(x) - f_i(x^t)|$ is large; when we don't, $f_(x^t)$

accurately estimates $f_i(x)$.

Recall from last time, these were the two conditions needed to prove that MW converged quickly. Then use MW convergence theorem to show that after $c = \frac{4\log|\mathcal{X}|}{\alpha^2}$ updates, PrivateMW answers all queries in Q approximately correctly. □

Let us now go through the proof more carefully.

Proof. We start by reminding the reader of the following accuracy guarantees of multiplicative weights and of SparseVector.

Theorem 7 (Sparse Vector Accuracy)**.** *For any sequence of k sensitivity-1 queries $\{f_1, f_2, \dots f_k\}$ satisfying $|\{i<k \mid f_i(x) \geq T - \alpha\}| < c$, SparseVector is (α, β)-numeric accurate for*

$$\alpha = \begin{cases} \frac{9c\left(\log k + \log\left(\frac{4c}{\beta}\right)\right)}{\epsilon}, & \text{if } \delta = 0 \\ \frac{9(\log k + \log(4c/\beta))\sqrt{8c\log(2/\delta)}}{\epsilon}, & \text{if } \delta > 0 \end{cases}.$$

Remember that (α, β)-accuracy requires that, with probability $1 - \beta$:

- We have $f_i(x) \leq T + \alpha$ for all queries f_i we did not answer
- We have $|f_i(x) - a_i| \leq \alpha$ for all queries f_i we answer, where a_i is the (noisy) numeric answer to our query.
- We do not terminate early. This means we answer at most c queries among the k we have seen so far. This follows from the fact that we only answer queries with $f_i(x) + Lap(1/\varepsilon) \geq \hat{T} = T + Lap(1/\varepsilon)$. In fact, in the proof of sparse vector, this is implied by the fact that we have no more than c queries with $f_i(x) \geq T - \alpha$, and concentration bounds on $\hat{T}$ and the Laplace noise we add (we argued $\hat{T}$ was $\alpha/2$-close to T, and similarly $|Lap(1/\varepsilon)| \leq \alpha/2$.).

Theorem 8. *Fix a class of linear queries Q and a database $x \in \Delta([|\mathcal{X}|])$, and let $x^1 \in \Delta([|\mathcal{X}|])$ be the uniform distribution over $\mathcal{X}$. That is, let $x_i^1 = \frac{1}{|\mathcal{X}|}$ for all i. Now consider a maximal length sequence of databases x^t for $t \in \{2, \dots, L\}$ generated by setting $x^{t+1} = MW(x^t, f_t, v_t)$ with learning rate $\eta = \alpha/2$, where for each t, $f_t \in Q$ and $v_t \in \mathbb{R}$,*

1. *$|f_t(x) - f_t(x^t)| > \alpha$ and*
2. *$|f_t(x) - v_t| < \alpha$.*

Then $L \leq \frac{4\log\mathcal{X}}{\alpha^2}$.

Now, the idea is then to pick $c = \frac{4\log|\mathcal{X}|}{\alpha^2}$: indeed, we are expecting to make at most c mistakes in which $|f_i(x^t) - f_i(x)|$ is large as long as x^t is chosen according to multiplicative weights. We will also pick $T = 2\alpha$. Also, we set $k = 2|Q|$, because we want to be able to use up to $2|Q|$ queries in SparseVector (we compare f'_{2i-1} and f'_{2i} to our threshold T for each query f_i). We can then plug the accuracy guarantees of both MW and SparseVector to get the

result. More precisely, in the $\delta = 0$ case, as long as (note that the $\|x\|_1$ renormalization comes from the fact that we are no longer working with sensitivity 1 queries as in SparseVector, but with sensitivity $1/\|x\|$ queries)

$$\alpha \geq \frac{9c\left(\log k + \log\left(4c/\beta\right)\right)}{\varepsilon\|x\|_1} = \frac{36\log|\mathcal{X}|\left(\log 2|Q| + \left(16\log\left(|\mathcal{X}|/\beta\alpha^2\right)\right)\right)}{\alpha^2\|x\|_1\epsilon}$$

we have that with probability at least $1-\beta$:

1. By the accuracy guarantee of Sparse Vector, we have that for all i such that we do not answer answer f_i, it must be that both $f'_{2i-1}(x) \leq T + \alpha = 3\alpha$ and $f'_{2i}(x) \leq T + \alpha = 3\alpha$. This can be rewritten immediately as

   $$|f_i(x) - f_i(x^t)| \leq 3\alpha,$$

   giving us the desired accuracy guarantee.

2. For all queries such that we have a numeric answer, i.e. either E_{2i-1} or E_{2i} is in $\mathbb{R}$. So, we have two cases:

   (a) $E_{2i-1} \in \mathbb{R}$: then, we have that E_{2i-1} must be close to $f_i(x) - f_i(x^t)$: the accuracy guarantee of SparseVector requires that

   $$|E_{2i-1} - f_{2i-1}(x)| = \left|E_{2i-1} - \left(f_i(x) - f_i(x^t)\right)\right| \leq \alpha.$$

   We then have that

   $$|a_i - f_i(x)| = |f_i(x^t) + E_{2i-1} - f_i(x)| \leq \alpha.$$

   (b) Else, we must have that $E_{2i} \in \mathbb{R}$. We then get the same result that $|a_i - f_i(x)| \leq \alpha$ by the same argument.

3. The algorithm does not halt early and we do not leave some of the queries un-answered, which means we have either $E_{2i-1} \in \mathbb{R}$ or $E_{2i} \in \mathbb{R}$ at most c times. This is implied by the fact that we have $f'_{2i-1}(x) \geq T - \alpha = \alpha$ or $f'_{2i}(x) \geq T - \alpha = \alpha$ at most c times.

   This last statement follows from the accuracy of the Multiplicative Weight algorithm, which states that there can be almost $\frac{4\log|X|}{\alpha^2} = c$ times that we have i) $|f'_{2i-1}(x)| = |f_i(x) - f_i(x^t)| \geq \alpha$ (or $|f_{2i}(x)| \geq \alpha$) and ii) $|a_i - f_i(x)| \leq \alpha$. This second condition is always satisfied when we have $E_j \in \mathbb{R}$, by point #2/the accuracy of SparseVector, i) holds at most c times.

To get the final accuracy guarantees, it just suffices to note that

$$\alpha \geq \frac{36\log|\mathcal{X}|\left(\log 2|Q| + \log\left(16\log\left(|\mathcal{X}|/\beta\alpha^2\right)\right)\right)}{\alpha^2\|x\|_1\epsilon}$$

is satisfied by the desired bound (will not do the exact algebra here). □