Addressing Budget Allocation and Revenue Allocation in Data Market Environments Using an Adaptive Sampling Algorithm

Boxin Zhao [1] **Boxiang Lyu** [1] **Raul Castro Fernandez** [2] **Mladen Kolar** [1]

Abstract

High-quality machine learning models are dependent on access to high-quality training data. When the data are not already available, it is tedious and costly to obtain them. *Data markets* help with identifying valuable training data: model consumers pay to train a model, the market uses that budget to identify data and train the model (the *budget allocation* problem), and finally the market compensates data providers according to their data contribution (*revenue allocation* problem). For example, a bank could pay the data market to access data from other financial institutions to train a fraud detection model. Compensating data contributors requires understanding data's contribution to the model; recent efforts to solve this revenue allocation problem based on the Shapley value are inefficient to lead to practical data markets. In this paper, we introduce a new algorithm to solve budget allocation and revenue allocation problems simultaneously in linear time. The new algorithm employs an adaptive sampling process that selects data from those providers who are contributing the most to the model. Better data means that the algorithm accesses those providers more often, and more frequent accesses corresponds to higher compensation. Furthermore, the algorithm can be deployed in both centralized and federated scenarios, boosting its applicability. We provide theoretical guarantees for the algorithm that show the budget is used efficiently and the properties of revenue allocation are similar to Shapley's. Finally, we conduct an empirical evaluation to show the performance of the algorithm in practical scenarios and when compared to other baselines. Overall, we believe that the new algo-

rithm paves the way for the implementation of practical data markets.

1. Introduction

The success of modern machine learning is largely dependent on access to high-quality data. However, in many scenarios, *consumers* of machine learning (ML) models do not have the necessary data to train them, and collecting data can be impractical; when it is practical it is often tedious and time-consuming. At the same time, data-rich organizations (*providers*) may not be aware that their data would be valuable to prospective consumers. For example, banks could improve fraud detection if they had access to relevant data from other banks with similar financial products. Data markets are a mechanism to address this problem. A well-functioning *data market* compensates *providers* for the data they supply, and allocates the data to those *consumers* (Fernandez et al., 2020) who need it, thus unleashing the value of data for everyone involved in the transaction. Designing such data markets remains an open problem.

Consider a group of 10 regional banks. One of them (the *consumer*) has detected unusual activity in transactions pertaining to a new credit card product. In response, ML engineers design a fraud detection model for the new product, but realize that they lack training data. These training data could be provided by the other 9 banks, the *providers*, in exchange for some compensation. Clearly, the consumer would like to invest their *budget* in high-quality data that will contribute to better models (the *budget allocation* problem). And providers would like to be compensated according to the value their data brings to the model (the *revenue allocation* problem); e.g., if a bank's data are more relevant to the task at hand, such bank should be compensated higher than the others. In this example, a well-functioning data market aligns the incentives of *consumers* with those of *providers*: it ensures that the budget is used towards obtaining high-quality data.

While there is growing literature on data markets (Fernandez et al., 2020; Liu et al., 2021; Agarwal et al., 2019; Li et al., 2021), no existing data market solves the *budget allocation* and *revenue allocation* problems simultaneously and

efficiently. For example, Li et al. (2021) deals with *budget allocation* but does not discuss *revenue allocation*. And there is a plethora of revenue allocation algorithms based on the Shapley value (Shapley, 1953) and approximations (Agarwal et al., 2019; Liu et al., 2021; Ghorbani & Zou, 2019; Jia et al., 2019b) that, although provide good guarantees on the *fairness* of the allocation, are too computationally expensive to lead to practical solutions.

In this paper, we introduce a new algorithm that solves simultaneously the budget and revenue allocation problems in linear time: much more efficiently than existing revenue allocation approaches. At each training iteration, the algorithm chooses a provider from whom to obtain data. The number of iterations is determined by the budget. Every time the algorithm accesses a provider's data, it compensates that provider using a fixed unit of consumer-provided budget. The key idea behind the algorithm is an ***Online Stochastic Mirror Descent (OSMD) sampler*** : an adaptive sampling procedure that ensures that the budget is spent on the providers that supply the best data. This means that providers with better data are accessed more often, and thus compensated higher. Most importantly, the new algorithm trains the model only once and has a complexity of $O(n \log n)$, with n being the number of providers. Contrast this with the Shapley value, which must consider the marginal contribution of one provider to all other provider subsets (Deng & Papadimitriou, 1994), leading to a complexity of $\mathcal{O}(2^n)$, and requiring a model training each time.

The key technical intuition behind the new algorithm is that it uses information contained in the optimization/training process to do both revenue allocation and budget allocation. While the Shapley value (and approximations) treats the optimization process as a black-box and only relies on the output model to do the revenue allocation, the new algorithm instead uses model updates to measure data quality after each iteration. This serves as the basis for the new adaptive sampling strategy that ensures that providers with a higher quality history will be accessed more often. By encouraging highly valuable providers to be accessed more, we obtain a better performing model, which then results in better budget allocation. Crucially, because the new algorithm does not require repeated training, it can be applied to large ML models that are costly to train, which is more practical than previous methods that require multiple re-trainings, and thus are often tested only on simple models, limiting their applicability.

The new algorithm is widely applicable. First, it supports any ML models that are trained in an iterative way. Second, it can be used in centralized and federated scenarios (Kairouz et al., 2021; McMahan et al., 2017). In centralized scenarios, a central data market platform stores data from providers and trains the model on behalf of consumers. In federated scenarios, data never leave the providers' premise, so it may be more adequate when data are sensitive or subject to regulations. The model is mainly trained by the providers, and the market platform is in charge of running the adaptive sampling algorithm and budget allocation. While in the first scenario, the cost of training the model is absorbed by the data market platform, in the second, it is distributed across providers. We do not deal in this paper with data market architectures or with mechanisms for determining how to use budget to compensate costs. Instead, we note that in both cases, reducing the cost of budget and revenue allocation strictly benefits all scenarios, and we focus on proposing an efficient algorithm.

Finally, we provide theoretical guarantees on both revenue allocation and budget allocation. For revenue allocation, we show that the new algorithm has properties similar to the four axioms of the Shapley value (Ghorbani & Zou, 2019; Jia et al., 2019a), i.e., efficiency, linearity, null player, and symmetry. For budget allocation, we show that the average regret of our method compared to spending all budget on the most valuable providers is asymptotically zero as the total budget goes to infinity. We complement the theoretical guarantees that make the algorithm principled with an extensive empirical evaluation that shows the practical applicability of the new method.

Contributions: We propose an algorithm that achieves revenue allocation and budget allocation at the same time. The proposed algorithm is highly-scalable and achieves similar revenue allocation quality as Shapley-based methods but more efficiently ($O(n \log n)$). Additionally, the new algorithm can be applied to federated learning settings and centralized scenarios seamlessly, increasing its applicability. From a technical perspective, the key technique of the algorithm is an adaptive sampling strategy that uses the information contained in the training process, which is generally ignored by previous research. We provide both theoretical and experimental justifications for our method.

To the best of our knowledge, this is the first highly-scalable method that unifies revenue allocation and budget allocation, thus paving the way to practical data markets.

Notation: For two positive sequences $(a_n)_{n \geq 1}$ and $(b_n)_{n \geq 1}$, we use $a_n = O(b_n)$ to denote that there exists a positive constant $C > 0$ such that $a_n \leq Cb_n$ when n is large enough. We use $a_n = o(b_n)$ when $\lim_{n \to \infty}(a_n/b_n) = 0$, and $a_n = o(1)$ when $\lim_{n \to \infty} a_n = 0$. For $n \geq 1$, we define $\mathcal{P}_{n-1} := \{x \in \mathbb{R}^n : \sum_{i=1}^n x_i = 1, x_i \geq 0 \text{ for all } i = 1, \ldots, n\}$ as the $(n-1)$-dimensional simplex.

Organization: In Section 2, we summarize the related work. In Section 3, we introduce our assumptions on the market. We then introduce our proposed method in Section 4, and

discuss its theoretical guarantees on budget allocation and revenue allocation in Section 4.1 and Section 4.2, respectively. The experimental results are in Section 5. We finally conclude our paper with Section 6. The code of the paper is available at: `https://github.com/boxinz17/Data-Market-via-Adaptive-Sampling`.

2. Related Work

There have been many studies on building a data market from various communities. Despite the vast literature and the growing need, there is no practical and principled data market. We summarize the literature most relevant to our work without attempting to be comprehensive. Fernandez et al. (2020) shares the authors' vision of the data market. Our project can be regarded as a special case of this vision, where we focus on the data market designed for machine learning applications. In addition, Agarwal et al. (2019); Li et al. (2021); Liu et al. (2021) discuss the market design for machine learning, but focus on different aspects from our paper. Chen et al. (2018); Kong et al. (2020); Zhang et al. (2021) offer complementary work on mechanism design.

Data pricing is a related topic with techniques that are complementary to our paper. Balazinska et al. (2011); Lin & Kifer (2014); Chawla et al. (2019); Aly et al. (2019); Chen et al. (2019) discuss pricing mechanisms for data, queries, and models in various problems. See Pei (2022); Fricker & Maksimov (2017); Liang et al. (2018) for recent surveys. One may use the pricing methods in the above literature to decide the price of each query of model updates from a data provider or to decide the total number of queries.

Data valuation is used for revenue allocation and, therefore, is closely related to our work (Koh & Liang, 2017; Sharchilev et al., 2018). A recent line of research reinterprets the data valuation problem as a cooperative game and applies the Shapley value (Shapley, 1953) to quantify the value of data (Ghorbani & Zou, 2019; Jia et al., 2019b;a; Ghorbani et al., 2020; Kwon et al., 2021; Kwon & Zou, 2022). In addition, Yoon et al. (2020) proposes a data valuation approach using reinforcement learning (RL). The major problem with the above approaches is that they are computationally costly, which restricts their application to simple ML models. Furthermore, the RL-based method needs access to the raw data, and thus is not applicable to privacy-sensitive scenarios such as federated learning.

Adaptive sampling via online learning approach has been applied in optimization literatures (Namkoong et al., 2017; Borsos et al., 2018; 2019; Hanchi & Stephens, 2020; Zhao et al., 2021; 2023), where people use it to construct unbiased estimators of the full gradient with an online learning objective to minimize the cumulative variance. In this paper, we adopt a similar idea but the objective is to maximize the cumulative utility gain. While some techniques are similar, this paper solves a fundamentally different problem.

Finally, several companies have attempted to create practical data markets, such as Scale AI (scale.ai, 2016), Dawex (Dawex, 2017) and Xignite (Xignite, 2017). Our method provides a principled algorithmic approach for practical data markets.

3. Preliminaries and Market Model

We start by formalizing the three parties involved in the data market: data consumers, data providers, and the market arbiter. We describe their roles and state assumptions on their behavior. Throughout the paper, we focus on serving one data consumer. We discuss extensions to serve multiple consumers in Section 6. Furthermore, the design supports a centralized model where providers' data is hosted by the arbiter and the arbiter is in charge of running computation. And it also supports a federated model, where providers' data stay with them and they are responsible for running computation on their own data.

Data Consumer has a machine learning model that needs to be trained. Let f_w denote the model parameterized by $w \in \mathbb{R}^d$. The data consumer provides a loss function l that can be used to find a suitable model parameter w by minimizing the loss $l(w; \mathcal{D})$, where $\mathcal{D}$ is a data set. The loss function is shared with the arbiter and, in the federated scenario, also with data providers. In addition to the loss function, the data consumer also provides a utility function $U : \mathbb{R}^d \mapsto [0, 1]$ that can be used to measure the quality of the model f_w. For example, the utility function can measure the quality of the produced model on a hold-out dataset that the data consumer has. We assume that the utility function is shared with the market arbiter but not with the data providers. Finally, we assume that the data consumer has a budget B, which is the total number of model updates that they can afford. Note that each model update costs the same amount of budget. We discuss extensions at the end of the section.

Data Providers are the market participants that have data that can be used to train machine learning models. We assume that there are n data providers. We do not make assumptions on the type of data the providers have. Instead, we assume that each data provider has a *model update oracle*. The ith data provider has an oracle function $\mathcal{O}_i : \mathbb{R}^d \mapsto \mathbb{R}^d$ that maps a model parameter w to an update $g_i = \mathcal{O}_i(w)$. Each time a data provider uses its model update oracle—locally in the federated scenario, or remotely, through an arbiter, in the centralized scenario—it will be compensated by one unit out of the total budget. There can be many options of $\mathcal{O}_i$. For example, $\mathcal{O}_i(w)$ might be the negative full gradient or stochastic gradient of the loss

function evaluated at w by using the data of provider i; or it can be $w_i^+ - w$, where w_i^+ is obtained by running several steps of stochastic gradient descent (SGD) starting from w; furthermore, the provider i may also choose some privacy protection techniques when computing $\mathcal{O}_i(w)$, such as differential privacy (Dwork & Roth, 2014; Chaudhuri et al., 2011). We leave the freedom of choosing $\mathcal{O}_i$'s to the data providers. Different choices of $\mathcal{O}_i$ will involve different levels of computational cost and privacy leakage risk. At one extreme, the data provider can return a random vector in $\mathbb{R}^d$ and thus induces little computational cost and privacy leakage. However, the design of our algorithm encourages the data provider to provide high-quality model update.

Market Arbiter maintains an iterative training process until the consumer's total budget is exhausted. Let K be the number of providers chosen at each round and we discuss the choice of K later in Section 4.1. Then the training process proceeds in $T = \lfloor B/K \rfloor$ iterations. Let w^0 and $p^0 = (1/n, \ldots, 1/n)^\top$ be the initial model parameter and sampling distribution, respectively. For iterations $t = 0, \ldots, T-1$, the market arbiter repeats the following process: (i) the arbiter uses the sampling distribution p^t to sample with replacement a set, $S^t \in [n]^K$, of K data providers;[1] (ii) the arbiter broadcasts the current model parameter w^t to the ith provider if $i \in S^t$;[2] (iii) the ith provider, $i \in S^t$, computes the model update $g_i^t = \mathcal{O}_i(w^t)$; (iv) the arbiter updates the sampling distribution based on the received marginal utilities $\{U(w^t + \gamma^t g_i^t)\}_{i \in S^t}$ to obtain p^{t+1}, where γ^t is the optimization stepsize; (v) the arbiter updates the model parameter as $w^{t+1} \leftarrow w^t + (\gamma^t/K) \sum_{i \in S^t} g_i^t$. Algorithm 1 details the process.

We detail the step (iv) in the next section. The training process executed the market arbiter is the same as the training process in federated learning (Kairouz et al., 2021) if $p^{t+1} = p^t$ in step (iv). Thus our method can be applied to the setting where providers may have privacy concerns or are physically distributed.

Finally, we note that each access costs the same fixed amount of budget. This makes sense because in the context of ML training, a single update does not have a major impact on the result. However, it is conceivable to introduce pricing techniques to price each access differently. We do not deal with this scenario in this paper.

---

[1] We choose sampling with replacement to obtain a clearer budget allocation analysis. The key intuition is that when using sampling with replacement, the providers in S^t are pairwise independent, allowing us to analyze them independently. In contrast, if we use sampling without replacement, we need to consider S^t as a whole, which is overly complicated given that there are $\binom{n}{K}$ total possible subsets.

[2] Here broadcast is a local operation in the central setting where providers' data is directly accessible to the market, and a distributed operation in the federated setting.

4. Online Stochastic Mirror Descent Sampler

We detail step (iv) of Algorithm 1 where the sampling distribution is updated. The sampling distribution is a crucial ingredient in the design of Algorithm 1 as it connects to budget and revenue allocation. We start by examining the expected utility gain in the round t. We would like to choose the set of providers S^t to maximize the utility gain

$$\Delta U^t := U\left(w^{t+1}\right) - U\left(w^t\right)$$
$$= U\left(w^t + \frac{\gamma^t}{K} \sum_{i \in S^t} g_i^t\right) - U\left(w^t\right).$$

Maximizing the utility gain over S^t is hard as it is a combinatorial problem. More specifically, the utility gain of selecting one provider depends not only on the provider itself but also on the other providers chosen in S^t. As a result, one needs to consider each subset of $[n]$, leading to a total of $\binom{n}{K}$ choices. The size of this space induces significant challenge.

To overcome this issue, one can consider the surrogate loss defined by *cumulative marginal utility gain*:

$$\Delta U_{\text{cm}}^t := \sum_{i \in S^t} U\left(w^t + \gamma^t g_i^t\right) - U\left(w^t\right). \qquad (1)$$

The advantage of marginal utility is that when selecting one provider, the utility gain depends solely on that provider and is independent of other providers. In this way, we can treat each provider independently, resulting in a total of n choices, which is much smaller than $\binom{n}{K}$ for $K > 1$. The surrogate loss simplifies the analysis and also leads to more efficient algorithm design. When $K = 1$, we have $\Delta U_{\text{cm}}^t = \Delta U^t$; when $K > 1$, a larger ΔU_{cm}^t also generally implies a larger ΔU^t.

Let $\Delta U_i^t = U(w^t + \gamma^t g_i^t) - U(w^t)$ and $p^t = (p_1^t, \ldots, p_n^t)^\top$. Recall that the set S^t is obtained by sampling with replacement from p^t and, thus, we have

$$\mathbb{E}_{S^t}\left[\Delta U_{\text{cm}}^t\right] = K \sum_{i=1}^n p_i^t \Delta U_i^t,$$

where $\mathbb{E}_{S^t}[\cdot]$ denotes the expectation taken with respect to S^t. Therefore, the distribution p^t can be chosen to maximize $\mathbb{E}_{S^t}[\Delta U_{\text{cm}}^t]$ when $\Delta U_1^t, \ldots, \Delta U_n^t$ are known.

In practice, obtaining $\Delta U_1^t, \ldots, \Delta U_n^t$ requires access to the model update oracle of each provider, which is too costly. Furthermore, since S^t is sampled based on p^t, the information will not be revealed until we make a decision on the sampling distribution. In each round, the arbiter has a partial information history $\{\{\Delta U_i^l\}_{i \in S^l}\}_{l=0}^{t-1}$ and $\{p^l\}_{l=0}^{t-1}$, based on which it updates the sampling distribution. We do not make any assumptions on how the utility is generated; thus

---

Algorithm 1 Adaptive Sampling with Online Stochastic Mirror Descent (OSMD) Sampler

---

1: **Input:** OSMD learning rate η; optimization stepsizes $\{\gamma_t\}_{t=0}^{T-1}$; parameter $\alpha \in [0,1]$, $\mathcal{A} = \mathcal{P}_{n-1} \cap [\alpha/n, \infty)^n$; total budget B, batch size K, and the number of communication rounds $T = \lfloor B/K \rfloor$; the initialization sampling distribution p^0 and model parameter w^0.
2: **Output:** Trained model parameter $\hat{w}$ and number of access for each data provider $N^T(i)$ for all $i \in [n]$.
3: Initialize $N^0(i) = 0$ for all $i \in [n]$.
4: **for** $t = 0, \ldots, T-1$ **do**
5:      Sample a batch of data providers $S^t = \{i^{t,0}, \ldots, i^{t,K-1}\} \in [n]^K$ with replacement by sampling distribution p^t.
6:      For each chosen data provider $i \in S^t$, broadcast the current model parameter w^t to it and receive the model update from the provider's model update oracle $g_i^t = \mathcal{O}_i(w^t)$.
7:      Let $N^{t+1}(i) = N^t(i) + \sum_{k=0}^{K-1} \mathbb{1}\{i^{t,k} = i\}$ for all $i \in [n]$.
8:      Let $\hat{u}^t = (\hat{u}_1^t, \ldots, \hat{u}_n^t)^\top$, where

$$\hat{u}_i^t = \frac{\sum_{k=0}^{K-1} \mathbb{1}\{i^{t,k} = i\}}{K p_i^t} \times \left(U\left(w^t + \gamma^t g_i^t\right) - U\left(w^t\right) \right)$$

9:      Solve $p^{t+1} = \mathrm{argmin}_{q \in \mathcal{A}} - \eta\langle q, \hat{u}^t \rangle + D_\Phi(q \,\|\, p^t)$ by Algorithm 2.
10:     Update w^t by $w^{t+1} = w^t + (\gamma^t/K)\sum_{i \in S^t} g_i^t$.
11: **end for**
12: Set $\hat{w} = w^T$.

Algorithm 2 OSMD Solver

1: **Input:** p^t, $\hat{u}^t$, η, $\alpha \in [0,1]$, and $\mathcal{A} = \mathcal{P}_{n-1} \cap [\alpha/n, \infty)^M$.
2: **Output:** p^{t+1}.
3: Let $\tilde{p}_i^{t+1} = p_i^t \exp(\eta \hat{u}_i^t)$ for $i \in [n]$.
4: Sort $\{\tilde{p}_i^{t+1}\}_{i=1}^n$ in a non-decreasing order: $\tilde{p}_{\pi(1)}^{t+1} \leq \ldots \leq \tilde{p}_{\pi(n)}^{t+1}$.
5: Let $v_i = \tilde{p}_{\pi(i)}^{t+1}\left(1 - \frac{i-1}{n}\alpha\right)$ for $i \in [n]$.
6: Let $u_i = \frac{\alpha}{n}\sum_{j=i}^n \tilde{p}_{\pi(j)}^{t+1}$ for $i \in [n]$.
7: Find the smallest i such that $v_i > u_i$, denoted as $i_\star^t$.
8: Let $p_i^{t+1} = \begin{cases} \alpha/n & \text{if } \pi^{-1}(i) < i_\star^t \\ \dfrac{\left((1-((i_\star^t-1)/n)\alpha)\tilde{p}_i^{t+1}\right)}{\left(\sum_{j=i_\star^t}^n \tilde{p}_{\pi(j)}^{t+1}\right)} & \text{otherwise.} \end{cases}$

---

it can be dependent on or even adversarial to the arbiter's decision.

The sampling process can be viewed as an online learning task with partial information feedback, where a game is played between the arbiter and the providers. The goal of the online learning task is to choose a sequence of sampling distributions to maximize $\mathbb{E}[\sum_{t=0}^{T-1} \Delta U_{\mathrm{cm}}^t]$. A key concern about this online learning task is the non-stationarity of the utility process $\{\Delta U_i^t\}_{t\geq 1}$, $i \in [n]$. More specifically, data providers that are useful in the early stage of the training process are not necessarily as useful in the later stage. To understand why, suppose that the model parameter is randomly initialized and far away from a good model parameter. At this stage, even a noisy data set can help improve the model by providing a rough guidance — the additional advantage of a high-quality data set may not be obvious at this stage as improving the model is easy. However, as the training

proceeds, the model will achieve reasonably good performance, and further improvement of the model will require higher quality data — the difference between the utilities of valuable and noisy providers now becomes more distinctive.

Based on the above discussion, we propose to use online stochastic mirror descent (OSMD) (Lattimore & Szepesvári, 2020, Chapter 31) to make update of the sampling distribution. Let $D_\Phi(x \,\|\, y) = \Phi(x) - \Phi(y) - \langle \nabla\Phi(y), x - y \rangle$ for $x, y \in \mathbb{R}_+^n$ be the Bregman divergence, where $\Phi(x) = \sum_{i=1}^n x_i \log x_i$ with $0 \log 0$ defined as 0, is the unnormalized negative entropy. Then the key updating rule is achieved by solving the following optimization problem:

$$p^{t+1} = \mathrm{argmin}_{q \in \mathcal{A}} - \eta\langle q, \hat{u}^t \rangle + D_\Phi(q \,\|\, p^t) \quad (2)$$

where p^t is the current sampling distribution, $\hat{u}^t$ is an unbiased estimate of the utility vectors $(\Delta U_1^t, \ldots, \Delta U_n^t)^\top$, and η is a chosen learning rate. We restrict the sampling distribution in a clipped simplex $\mathcal{A} = \mathcal{P}_{n-1} \cap [\alpha/n, \infty)^n$, where

$\alpha \in [0, 1]$ is a tuning parameter. By restricting the sampling distribution to be away from zero, we avoid the algorithm from committing too hard on a single provider, which otherwise may prevent the discovery of change points in a non-stationary environment. The first part of the objective encourages the sampling distribution to maximize the most recent utility gain, while the second part of the objective prevents it from deviating too far away from the previous sampling distribution. The detailed algorithm is described in Algorithm 1. The problem in (2) can be solved efficiently by Algorithm 2 (Proposition B.1 in Appendix B.1). The most costly step of Algorithm 2 is sorting $\{\tilde{p}_i^{t+1}\}_{i=1}^n$, which has a computational complexity of $O(n \log n)$. However, noting that most of the entries in $\tilde{p}^{t+1}$ are sorted in the previous round, we may use an adaptive sorting algorithm to achieve a much faster running time (Estivill-Castro & Wood, 1992).

4.1. Analysis of Budget Allocation

We provide theoretical guarantees on budget allocation in the form of a regret bound when comparing against an oracle competitor. At the beginning of round t, we allow the competitor to know the provider with the largest ΔU_i^t; thus, to maximize ΔU_{cm}^t, the competitor will choose it K times in round t. Due to the non-stationary nature of the problem, we also allow the oracle to change its decision from iteration to iteration, with the total number of changes not exceeding $m - 1$ times, where m is a chosen tuning parameter. Note that with a larger m, we are competing against a more flexible and harder oracle.

For $m \geq 1$, the action space Γ_m is defined as

$$\Gamma_m := \Big\{ \big(a^0, \ldots, a^{T-1}\big) \in [n]^T : $$
$$\sum_{t=1}^{T-1} \mathbb{1}\left\{a^t \neq a^{t-1}\right\} \leq m - 1 \Big\}.$$

The space Γ_m contains all action sequences where the action changes at most $m - 1$ times. This space contains the action sequence of an oracle that knows the identities of the top m providers and the right order to access them. We define regret of the budget allocation of Algorithm 1 as

$$R_B := \mathbb{E}\left[\max_{(a^t) \in \Gamma_m} K \sum_{t=0}^{T-1} \Delta U_{a^t}^t - \sum_{t=0}^{T-1} \Delta U_{\text{cm}}^t\right], \quad (3)$$

where ΔU_{cm}^t is defined in (1) and the expectation is taken with respect to all sources of randomness. From the consumer's perspective, a larger utility gain means a better budget allocation and, therefore, regret R_B measures the loss of budget due to the lack of quality information about providers. We have the following guarantee on the regret of Algorithm 1.

Theorem 4.1. *Assume $B = KT$ and $U(\cdot) \in [0, 1]$. Let $\{p^t\}_{t=0}^{T-1}$ be the sequence of sampling distribution generated by Algorithm 1. The regret R_B defined in (3) is bounded by*

$$R_B \leq \alpha B + \frac{\eta n B}{2} + \frac{mK \log(n/\alpha)}{\eta}.$$

If $\alpha = \sqrt{n/B}$ and $\eta = \sqrt{m \log(nB)/(nT)}$, it follows that $R_B/B = O\left(\sqrt{nmK \log(nB)/B}\right)$. Treating n, K, m as constants, we have that the asymptotic average regret is zero, $R_B/B = o(1)$, which is used as a standard to measure budget allocation (Agarwal et al., 2019). By Theorem 4.1, we see that a larger K will induce a larger regret within the same budget; however, a larger K is helpful in reducing the variance of the model parameter. Thus, a good choice of K should maintain a trade-off between these two concerns. Note that we do not provide specific quantification of how the variance depends on K. The reason is that the variance of the model parameter depends on numerous factors, such as the loss function, the learning rate, and the model update oracle. We intentionally leave these factors unspecified to maintain generality. That being said, a simple example for reference is the use of mini-batch SGD or local SGD as the model update oracle. In this case, the convergence analysis from (Karimireddy et al., 2020) can be adopted. It then becomes evident that the number of providers we select in each round will influence the convergence speed through the variance.

4.2. Analysis of Revenue Allocation

Data providers furnish their data with the expectation of receiving compensation, thus the equitable distribution of revenue among providers is a crucial consideration for data markets. This differs from the budget allocation issue, which we framed as an online optimization problem. Revenue allocation, on the other hand, falls within the scope of a "fair division" problem (Fair division, 2018). Generally, there is no universally acknowledged standard for resolving revenue division dilemmas, as opposed to online optimization scenarios. A prevalent strategy is to first establish each participant's "entitlement" and subsequently distribute the revenue in accordance with this assigned entitlement.

Existing research primarily applies the Shapley value (Shapley, 1953) to determine entitlement, given its attributes that are considered essential for equitable allocation in cooperative games. The Shapley value is widely acknowledged to singularly satisfy four axioms of a fair reward distribution in cooperative game theory, namely: symmetry, efficiency, null player, and linearity (Shapley, 1953; Kwon & Zou, 2022). Nevertheless, the computation of the Shapley value is considerably resource-intensive. Consequently, the total funds available for allocation are reduced by the computational expense incurred in deriving the Shapley value itself. Furthermore, the relevance and necessity of these axioms for revenue allocation within real-world data markets remain a

matter of debate.

In this paper, we introduce a novel concept of entitlement referred to as the *number of accesses to the model update oracle*. By correlating the compensation a data provider receives with the quantity of model update oracles it contributes, the compensation aligns with the level of service provided. We posit that this methodology offers a fresh perspective on the study of revenue allocation.

Given that our model does not frame revenue allocation as a cooperative game, our approach diverges from the typical analytical framework associated with the Shapley value. Nevertheless, we demonstrate that our revenue allocation method possesses certain attributes that echo the underlying principles of the Shapley value's axioms.

- Symmetry: The symmetry axiom basically requires the evaluation result is invariant to random shuffling of the providers. Note that by the design of Algorithm 1, it is clear that the distribution of $N^T(i)$ for any $i \in [n]$ will not change by shuffling the providers, thus our revenue allocation approach enjoys the symmetry property.

- Efficiency: Efficiency axiom requires that the sum of values of all providers should be equal to the value of the whole dataset. In our case, by the definition of $N^T(i)$, we have $\sum_{i=1}^{n} N^T(i)$ equal the budget for training. Thus, our revenue allocation approach also shares certain efficiency property.

- Null Player: This axiom requires to assign zero value to data providers that make no contribution. In our case, any data provider has a chance to get compensation no matter how much contribution it makes. However, our revenue allocation approach ensures that providers making no contribution will get asymptotically minimum compensation. To see this, assuming that there exists an $i \in [n]$, such that $U(w + \gamma \mathcal{O}_i(w)) \leq U(w)$ for any $w \in \mathbb{R}^d$ and $\gamma > 0$ (which indicates that provider i cannot make positive contribution), then by the design of Algorithm 1, we will have $p_i^1 \geq p_i^2 \geq \ldots \geq \alpha/n$. This means that the probability of accessing provider i is asymptotically decreasing to the lower bound. When α is small, the compensation received by provider i will be negligible.

By the intrinsic design of our algorithm, it is hard for our revenue allocation approach to satisfy the linearity axiom; however, it is unclear why linearity matters in practical applications. In fact, previous research also tries to remove some of the four axioms for a more practical solution. For example, Kwon & Zou (2022) removes the efficiency axiom and proposes a variant of Shapley value which has better performance on many practical machine learning tasks. Here we relax the linearity axiom for a similar reason.
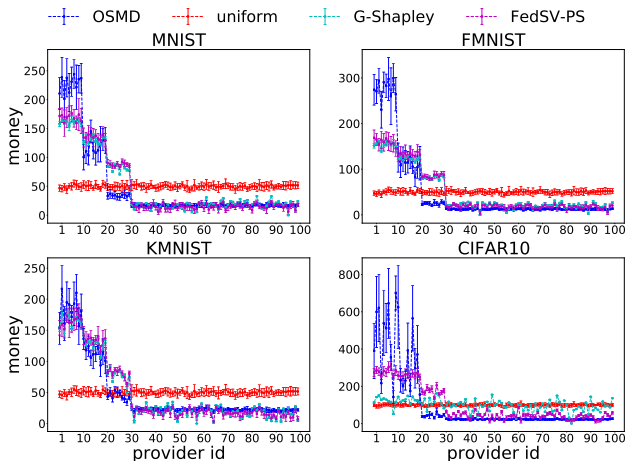

Figure 1. Revenue allocation result of OSMD (our method), uniform sampling, Gradient Shapley (G-Shapley) and Federated Shapley value with permutation sampling estimation (FedSV-PS). The solid dot denotes the mean and the error bar denotes the mean $\pm$ standard deviation.

5. Evaluation

We evaluate our method empirically on both real-world data and synthetic data experiments. We concentrate on three main questions. First, for revenue allocation, we ask whether our method provides similar allocation quality as the state-of-the-art methods based on Shapley value. For this purpose, we choose two Shapley value based methods as the baseline methods: Gradient Shapley (G-Shapley) (Ghorbani & Zou, 2019) and Federated Shapley value with permutation sampling estimation (FedSV-PS) (Wang et al., 2020). These two methods do not require re-training of the model multiple times, as required by most of the Shapley value based methods, and are suitable for modern large-scale machine learning models like Deep Neural Networks (DNN). Second, for budget allocation, we ask whether our method delivers better performing model than the standard training procedure based on uniform sampling. Finally, we study the runtime of our method. We compare it with uniform sampling, G-Shapley and FedSV-PS on real-world data sets.

5.1. Main Results

We implement the experiments on four image classification task data sets: MNIST (LeCun & Cortes, 2010), Fashion-MNIST (FMNIST) (Xiao et al., 2017), Kuzushiji-MNIST (KMNIST) (Clanuwat et al., 2018), and CIFAR-10 (Krizhevsky & Hinton, 2009). We create 100 data providers with each provider having 400 training samples. We then use another 1000 samples from the training set as the hold-out set to compute the utility, which is defined as the negative loss on the hold-out set. Finally, we use the

testing set which contains $10,000$ samples to measure the accuracy of the model. See Appendix A.1 for more details about data preparation. For MNIST, FMNIST and KMNIST, we train a two-layer neural network with one-hidden layer of 300 neurons; for CIFAR-10, we train a Convolutional Neural Network (CNN) with one convolutional layer and two feed-forward layers. Each method is repeated 10 times with different random seeds.

To create quality discrepancies across providers, we corrupt $\beta\%$ of the providers' samples by changing a sample's correct label to a randomly chosen wrong label. For provider 1–10, $\beta = 0$; for provider 11–20, $\beta = 20$; for provider 21–30, $\beta = 50$; for the rest of the providers, $\beta = 90$. Additional details are provided in Appendix A.1.

Revenue Allocation: We first examine whether our method achieves similar revenue allocation results as Shapley-based mehtods (we will show later it solves the problem much faster). For our method and uniform sampling, the revenue is allocated proportional to the number of accesses, while for G-Shapley and FedSV-PS, the revenue is allocated proportional to the positive part of Shapley value, defined as $\max\{\text{SV}, 0\}$, where SV denotes the Shapley value. The result is shown in Figure 1. We see that OSMD has a similar revenue allocation performance as G-Shapley and FedSV-PS on MNIST, KMNIST and FMNIST. OSMD tends to allocate more revenue on the most valuable providers while G-Shapley and FedSV-PS seem to keep a balance between different quality providers. Interestingly, we see that G-Shapley underperforms on CIFAR-10. This might be because the task is difficult, and with a high portion of data being corrupted, the one-step gradient approximation does not work well. The take-away message is that the new method's allocation closely resembles that of Shapley-based methods.

Budget Allocation: We study the performance of budget allocation. We show the model accuracy on the hold-out test set along the training process. The results in Figure 2 show that OSMD successfully helps the model obtain a better performance than uniform sampling by identifying the valuable providers and adjusting the sampling distribution accordingly: the new algorithm uses the budget efficiently.

Computational Cost: We also compare the runtime (wall clock time) of the four methods mentioned above. Higher runtimes imply higher costs on the arbiter side, thus reducing runtime is a crucial concern in designing practical data market, as the higher the costs, the higher the proportion of budget that needs to be used to cover those costs and thus the worse the models delivered to consumers. To measure the scalability of different methods, we fix the number of samples that each provider has to 100 and increase the number of providers from 50 to 400. The result is shown in Figure 3. Compared with G-Shapley and FedSV-PS, our
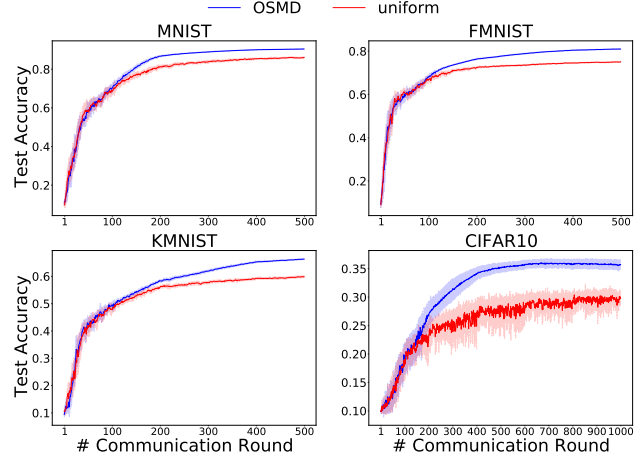

Figure 2. Budget allocation result of OSMD (our method) and uniform sampling. The solid line denotes the mean and the shadow region denotes the mean $\pm$ standard deviation.
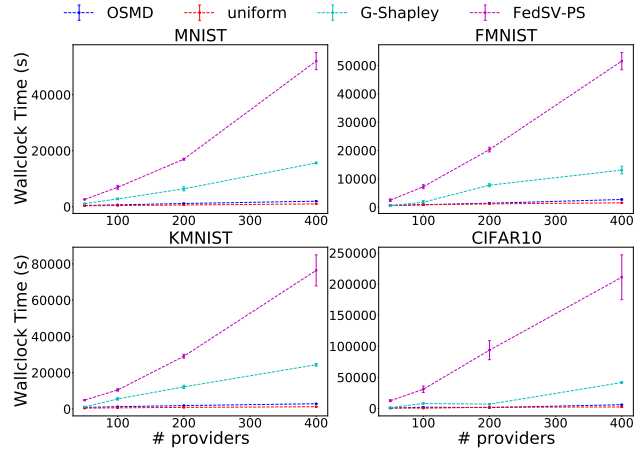

Figure 3. Runtime of OSMD (our method), uniform sampling, Gradient Shapley (G-Shapley) and Federated Shapley value with permutation sampling estimation (FedSV-PS). The solid dot denotes the mean and the error bar denotes the mean $\pm$ standard deviation.

method is much faster, only marginally more costly than uniform sampling and highly scalable. This result indicates that our method is more suitable to modern ML applications where the size of both dataset and model is large.

5.2. Mixture Linear Regression

In the previous section, the difference of data quality between providers is created by corrupting the label. In this section, we present a different approach to create such difference by creating distribution shift of $\mathbb{P}(y \mid x)$. More specifically, we implement a mixture linear regression ex-

peiment. Assume that data consumer wants to train a linear regression model. Both data consumer's and data providers' features are drawn from the same distribution $\mathbb{P}_X$; however, the response variable is generated by the following mixture linear regression model:

$$x_{ij} \in \mathbb{R}^d \sim \mathbb{P}_X, \quad j = 1, \ldots, m_i, i = 0, 1, \ldots, n,$$

$$z_0, z_1, \ldots, z_n \sim \text{uniform}([K]),$$

$$y_{ij} = \langle w_{z_i}, x_{ij} \rangle + \varepsilon_{ij}, \; \varepsilon_{ij} \overset{i.i.d.}{\sim} N(0, 0.5^2),$$
$$j = 1, \ldots, m_i, \; i = 0, 1, \ldots, n,$$

where z_0 is the latent group label of the consumer, and z_i with $i \geq 1$ is the latent group label of data provider i. The set of regression parameters is $\{w_1, \ldots, w_K\}$. This way, data providers with $z_i = z_0$ should have the same true parameter as the data consumer, thus is more valuable. We set $K = 4$ and $z_0 = 1$. We let $d = 1000$ and generate each entry of w_k i.i.d. from $U[0.5(k-1), 0.5k]$ for $k = 1, \ldots, 4$. In addition, we generate entries of x_{ij} i.i.d. from $N(0, 1)$ and we initialize the parameter from $(-1, \ldots, -1)^\top$. See details in Appendix A.2.

We compare the budget allocation and revenue allocation of our method with other baselines. The results are shown in Figure 4. Regarding budget allocation, we see that both OSMD and FedSV-PS generate fair revenue allocation, by which we mean that data providers with smaller $\|w_{z_i} - w_{z_0}\|$ are compensated higher. Interestingly, we see that the allocation result of G-Shapley is opposite to the truth. We conjecture this is because providers with larger $\|w_{z_i} - w_{z_0}\|$ will make gradients evaluated at the initial point have larger norm, and thus achieve more utility gain at the initial stage of the training. This is another example in addition to the previous CIFAR-10 case where G-Shapley fails. Regarding budget allocation, we see that there is a clear phase transition. Initially, since providers from all groups help, the parameter estimates get closer to the true parameter. All providers seem to have similar values, thus OSMD and uniform sampling perform similarly. As the parameter estimation gets closer to the true parameter, there is an increasing difference between providers, and OSMD starts to tell apart the "right" providers from the "wrong" providers. This experiment demonstrates the point raised in Section 4 about non-stationarity: that providers whose data is useful at a point in time may not be useful later. The take-away message is that taking the non-stationarity of the problem into consideration is necessary for the algorithm to perform well.

6. Conclusion

We proposed a new algorithm based on adaptive sampling that solves the problems of budget allocation and revenue allocation in linear time, thus paving the way to the implementation of practical data markets. The new algorithm
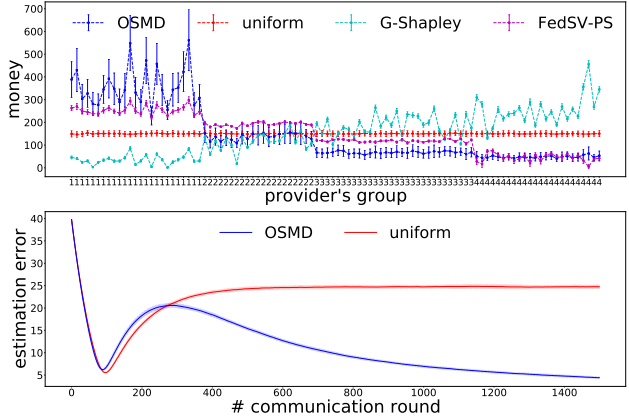

Figure 4. The result of mixture linear regression experiment. The above figure shows the revenue allocation result. Note that the consumer's parameter is from group 1. The below figure shows the budget allocation result.

achieves state-of-the-art revenue allocation performance while ensuring efficient budget allocation performance at the same time. The key insight of the algorithm design is to open the black box of the training process, and use an adaptive sampling technique to encourage more valuable providers to make more contributions. We offer theoretical guarantees and an empirical evaluation of the algorithm.

There are many interesting future directions. First, the problem of pricing each query of model update remains open. Second, while our algorithm can be easily extended to serve multiple consumers, it is worth noting that in practice, different consumers may cooperate or compete with each other. For example, the value of a machine learning model possessed by a company may not only depend on its own accuracy, but may also depend on the performance of the model possessed by the company's competitor. The study of such interactions between multiple consumers provides a fruitful research direction. In addition, one may also consider the effect of strategic behavior, especially if providers can collude. Finally, combining differential privacy with our method can also be a promising direction.

Acknowledgements

9

References

Agarwal, A., Dahleh, M. A., and Sarkar, T. A marketplace for data: An algorithmic solution. In *ACM Conference on Economics and Computation*, 2019.

Aly, H., Krumm, J., Ranade, G., and Horvitz, E. To buy or not to buy: Computing value of spatiotemporal information. *ACM Transactions on Spatial Algorithms and Systems*, 5(4):22:1–22:25, 2019.

Balazinska, M., Howe, B., and Suciu, D. Data markets in the cloud: An opportunity for the database community. *Proceedings of the VLDB Endowment*, 4(12):1482–1485, 2011.

Borsos, Z., Krause, A., and Levy, K. Y. Online variance reduction for stochastic optimization. In *Conference On Learning Theory*, 2018.

Borsos, Z., Curi, S., Levy, K. Y., and Krause, A. Online variance reduction with mixtures. In *International Conference on Machine Learning*, 2019.

Boyd, S., Boyd, S. P., and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Chaudhuri, K., Monteleoni, C., and Sarwate, A. D. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12:1069–1109, 2011.

Chawla, S., Deep, S., Koutris, P., and Teng, Y. Revenue maximization for query pricing. *Proceedings of the VLDB Endowment*, 13(1):1–14, 2019.

Chen, L., Koutris, P., and Kumar, A. Towards model-based pricing for machine learning in a data marketplace. In *International Conference on Management of Data*, 2019.

Chen, Y., Immorlica, N., Lucier, B., Syrgkanis, V., and Ziani, J. Optimal data acquisition for statistical estimation. In *ACM Conference on Economics and Computation*, 2018.

Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., and Ha, D. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

Dawex. https://www.dawex.com, 2017. Accessed: 2023-04-30.

Deng, X. and Papadimitriou, C. H. On the complexity of cooperative solution concepts. *Mathematics of operations research*, 19(2):257–266, 1994.

Dwork, C. and Roth, A. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3-4):211–407, 2014.

Estivill-Castro, V. and Wood, D. A survey of adaptive sorting algorithms. *ACM Computing Surveys*, 24(4):441–476, 1992.

Fair division. Fair division — Wikipedia, the free encyclopedia, 2018. URL https://en.wikipedia.org/wiki/Fair_division. [Online; accessed 30-April-2023].

Fernandez, R. C., Subramaniam, P., and Franklin, M. J. Data market platforms: Trading data assets to solve data problems. *Proceedings of the VLDB Endowment*, 13(11):1933–1947, 2020.

Fricker, S. A. and Maksimov, Y. V. Pricing of data products in data marketplaces. In *Software Business*, 2017.

Ghorbani, A. and Zou, J. Y. Data shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning*, 2019.

Ghorbani, A., Kim, M. P., and Zou, J. A distributional framework for data valuation. In *International Conference on Machine Learning*, 2020.

Hanchi, A. E. and Stephens, D. A. Adaptive importance sampling for finite-sum optimization and sampling with decreasing step-sizes. In *Advances in Neural Information Processing Systems*, 2020.

Jia, R., Dao, D., Wang, B., Hubis, F. A., Gürel, N. M., Li, B., Zhang, C., Spanos, C. J., and Song, D. Efficient task-specific data valuation for nearest neighbor algorithms. *Proceedings of the VLDB Endowment*, 12(11):1610–1623, 2019a.

Jia, R., Dao, D., Wang, B., Hubis, F. A., Hynes, N., Gürel, N. M., Li, B., Zhang, C., Song, D., and Spanos, C. J. Towards efficient data valuation based on the shapley value. In *International Conference on Artificial Intelligence and Statistics*, 2019b.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K. A., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Eichner, H., Rouayheb, S. E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konečný, J., Korolova, A., Koushanfar, F., Koyejo, S., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, R., Qi, H., Ramage, D., Raskar, R., Raykova, M., Song, D., Song, W., Stich, S. U., Sun, Z., Suresh, A. T., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F. X., Yu, H., and Zhao, S. Advances and open problems in federated learning. *Found. Trends Mach. Learn.*, 14(1-2):1–210, 2021.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., and Suresh, A. T. SCAFFOLD: stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, 2020.

Koh, P. W. and Liang, P. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, 2017.

Kong, Y., Schoenebeck, G., Tao, B., and Yu, F. Information elicitation mechanisms for statistical estimation. In *AAAI Conference on Artificial Intelligence*, 2020.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Kwon, Y. and Zou, J. Beta shapley: a unified and noise-reduced data valuation framework for machine learning. In *International Conference on Artificial Intelligence and Statistics*, 2022.

Kwon, Y., Rivas, M. A., and Zou, J. Efficient computation and analysis of distributional shapley values. In *International Conference on Artificial Intelligence and Statistics*, 2021.

Lattimore, T. and Szepesvári, C. *Bandit algorithms*. Cambridge University Press, 2020.

LeCun, Y. and Cortes, C. MNIST handwritten digit database. Technical report, New York University, Google Labs and Microsoft Research, 2010.

Li, Y., Yu, X., and Koudas, N. Data acquisition for improving machine learning models. *Proceedings of the VLDB Endowment*, 14(10):1832–1844, 2021.

Liang, F., Yu, W., An, D., Yang, Q., Fu, X., and Zhao, W. A survey on big data market: Pricing, trading and protection. *IEEE Access*, 6:15132–15154, 2018.

Lin, B. and Kifer, D. On arbitrage-free pricing for general data queries. *Proceedings of the VLDB Endowment*, 7(9): 757–768, 2014.

Liu, J., Lou, J., Liu, J., Xiong, L., Pei, J., and Sun, J. Dealer: An end-to-end model marketplace with differential privacy. *Proceedings of the VLDB Endowment*, 14(6):957–969, 2021.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *International Conference on Artificial Intelligence and Statistics*, 2017.

Namkoong, H., Sinha, A., Yadlowsky, S., and Duchi, J. C. Adaptive sampling probabilities for non-smooth optimization. In *International Conference on Machine Learning*, 2017.

Pei, J. A survey on data pricing: From economics to data science. *IEEE Transactions on knowledge and Data Engineering*, 34(10):4586–4608, 2022.

scale.ai. https://scale.com, 2016. Accessed: 2023-04-30.

Shapley, L. S. A value for n-person games. *Contributions to the Theory of Games*, pp. 2(28):307–317, 1953.

Sharchilev, B., Ustinovskiy, Y., Serdyukov, P., and de Rijke, M. Finding influential training samples for gradient boosted decision trees. In *International Conference on Machine Learning*, 2018.

Wang, T., Rausch, J., Zhang, C., Jia, R., and Song, D. A principled approach to data valuation for federated learning. In *Federated Learning - Privacy and Incentive*, volume 12500, pp. 153–167. Springer, 2020.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Xignite. https://www.xignite.com, 2017. Accessed: 2023-04-30.

Yoon, J., Arik, S. Ö., and Pfister, T. Data valuation using reinforcement learning. In *International Conference on Machine Learning*, 2020.

Zhang, M., Arafa, A., Wei, E., and Berry, R. Optimal and quantized mechanism design for fresh data acquisition. *IEEE Journal on Selected Areas in Communications*, 39 (5):1226–1239, 2021.

Zhao, B., Liu, Z., Chen, C., Kolar, M., Zhang, Z., and Zhou, J. Adaptive client sampling in federated learning via online learning with bandit feedback. *arXiv preprint arXiv:2112.14332*, 2021.

Zhao, B., Lyu, B., and Kolar, M. L-SVRG and L-Katyusha with adaptive sampling. *Transactions on Machine Learning Research*, 2023.

A. Details about Experiments

A.1. Details about Real-World Experiments

Dataset: We implement the experiments on four image classification task datasets: MNIST (LeCun & Cortes, 2010), Fashion-MNIST (FMNIST) (Xiao et al., 2017), Kuzushiji-MNIST (KMNIST) (Clanuwat et al., 2018), and CIFAR-10 (Krizhevsky & Hinton, 2009). All datasets are composed of training set and testing set. The sample size information is summarized in Table 1. Both training set and testing set of all datasets are perfectly balanced, that is, each label contains the equal number of samples. We create 100 data providers with each provider having 400 training samples. We then use 1000 samples from the rest of the training set as the hold-out set to compute the utility, which is defined as the negative loss on the validation set. Finally, we use the testing set to measure the accuracy of the model.

Dataset	Sample Size of Training Set	Sample Size of Testing Set	Label Balance
MNIST	$60,000$	$10,000$	Balanced
KMNIST	$60,000$	$10,000$	Balanced
FMNIST	$60,000$	$10,000$	Balanced
CIFAR10	$50,000$	$10,000$	Balanced

Table 1. The information about image classification datasets.

Machine Learning Model: For MNIST, FMNIST and KMNIST, we train a two-layer neural network with one-hidden layer of 300 neurons; for CIFAR-10, we train a Convolutional Neural Network (CNN) with one convolutional layer and two feed-forward layers. We use the cross-entropy loss as the loss function.

Model Update Oracle: For a given model parameter w, we choose $\mathcal{O}_i(w) = w_i^+ - w$, where w_i^+ is obtained by running mini-batch SGD on the provider i's data for one epoch. More specifically, we divide the provider's data into mini batches, that is, $\mathcal{D}_i = \cup_{l=1}^{L} \mathcal{B}_l$, and then let $w^0 = w$, $w^{l+1} = w^l - (\gamma_{\text{local}}/B)\nabla l(w^l, \mathcal{B}_l)$ for $l = 0, \ldots, L-1$, where B is the size of the mini-batches and γ_{local} is the local step size. Finally, we let $w_i^+ = w^L$.

Parameter Choice: We choose the optimization stepsizes $\{\gamma_t\}_{t=0}^{T-1}$ in Algorithm 1 as $\gamma^t = \gamma_{\text{global}} = 0.1$ for all $t = 0, \ldots, T-1$. Besides, we let $\alpha = 0.01$, and $\eta = 1.0$. We set the total number of communication rounds between the arbiter and providers as $T = 500$ for MNIST, FMNIST and KMNIST, and as $T = 1000$ for CIFAR10. Furthermore, we have $K = 0.1n$. For G-Shapley, we choose the same convergence criterion as in Ghorbani & Zou (2019); however, we set the back check length to be 30 for CIFAR-10 and to be 10 for the rest of tasks, and we set the convergence threshold as 0.01. For FedSV-PS, we use the same convergence criterion to decide the determination in each communication round. The back check length is 10 and the convergence threshold is 0.05. In addition, each method is repeated 10 times with different random seeds.

Corruption: To create quality discrepancies across providers, we corrupt $\beta\%$ of the providers' samples by changing a sample's correct label to a randomly chosen wrong label. For provider 1–10, $\beta = 0$; for provider 11–20, $\beta = 20$; for provider 21–30, $\beta = 50$; for the rest of the providers, $\beta = 90$.

Computational Cost Experiment: To compare the computational cost of our method with other competitors, we fix the number of training samples possessed by each provider as 100, and set the number of providers as 50, 100, 200 and 400, and record the runtime.

A.2. Details about Mixture Linear Regression Experiments

Data Generation Process: Both data consumer's and data providers' features are drawn from the same distribution $\mathbb{P}_X$; however, the response variable is generated by the following mixture linear regression model:

$$x_{ij} \in \mathbb{R}^d \sim \mathbb{P}_X \quad \text{for all } j = 1, \ldots, m_i, i = 0, 1, \ldots, n,$$
$$z_0, z_1, \ldots, z_n \sim [K] \text{ uniformly random,}$$
$$y_{ij} = \langle w_{z_i}, x_{ij} \rangle + \varepsilon_{ij}, \varepsilon_{ij} \overset{i.i.d.}{\sim} N(0, 0.5^2) \quad \text{for all } j = 1, \ldots, m_i, i = 0, 1, \ldots, n,$$

where z_0 is the latent group label of the consumer, and z_i with $i \geq 1$ is the latent group label of data provider i. The set of regression parameters is $\{w_1, \ldots, w_K\}$. This way, data providers with $z_i = z_0$ should have the same true parameter as the

data consumer, thus is more valuable. We set $K = 4$ and $z_0 = 1$. Besides, we let $d = 1000$ and generate each entry of w_k i.i.d. from $U[0.5(k-1), 0.5k]$ for $k = 1, \ldots, 4$. In addition, we generate entries of x_{ij} i.i.d. from $N(0, 1)$ and we initialize the parameter from $(-1, \ldots, -1)^\top$. Given a parameter estimation $\hat{w}$, the estimation error is measured by $\|\hat{w} - w_1\|$.

Training Process: For model update oracle, we let $\mathcal{O}_i(w) = -\nabla l(w, \mathcal{D}_i)$, where $\mathcal{D}_i = \{(x_{ij}, y_{ij})\}_{j=1}^{m_i}$ and

$$l(w, \mathcal{D}_i) = \frac{1}{2m_i} \sum_{j=1}^{m_i} (y_{ij} - \langle w, x_{ij} \rangle)^2 .$$

Besides, we choose the optimization stepsizes $\{\gamma_t\}_{t=0}^{T-1}$ in Algorithm 1 as $\gamma^t = \gamma_{\text{global}} = 0.01$ for all $t = 0, \ldots, T-1$. Besides, we let $\alpha = 0.01$, and $\eta = 0.001$. We set the total number of communication rounds between the arbiter and providers as $T = 1500$. Furthermore, we have $K = 0.1n$. The parameter setting of G-Shapley and FedSV-PS are the same as in Section A.1.

B. Technical proofs

B.1. Efficient Solution to (2)

The following proposition justifies Algorithm 2.

Proposition B.1. *Let*
$$\tilde{p}_i^{t+1} = p_i^t \exp\left(\eta \hat{u}_i^t\right), \qquad i \in [n].$$
Let $\pi : [n] \mapsto [n]$ *be a permutation such that* $\tilde{p}_{\pi(1)}^{t+1} \leq \cdots \leq \tilde{p}_{\pi(n)}^{t+1}$. *Let* $i_\star^t$ *be the smallest integer* i *such that*

$$\tilde{p}_{\pi(i)}^{t+1}\left(1 - \frac{i-1}{n}\alpha\right) > \frac{\alpha}{n}\sum_{j=i}^{n}\tilde{p}_{\pi(j)}^{t+1}.$$

Then the solution of (2) is

$$p_i^{t+1} = \begin{cases} \alpha/n & \text{if } \pi^{-1}(i) < i_\star^t \\ \left((1 - ((i_\star^t - 1)/n)\alpha)\tilde{p}_i^{t+1}\right) / \left(\sum_{j=i_\star^t}^{M} \tilde{p}_{\pi(j)}^{t+1}\right) & \text{otherwise}. \end{cases}$$

Proof. First, we show that the solution of (2), p^{t+1}, can be found as

$$\check{p}^{t+1} = \arg\min_{q \in \mathbb{R}_+^n} -\eta\langle q, \hat{u}^t \rangle + D_\Phi(q \| p^t),$$
$$p^{t+1} = \arg\min_{q \in \mathcal{A}} D_\Phi(q \| \check{p}^{t+1})$$

The optimality condition for $\check{p}^{t+1}$ implies that

$$-\eta\hat{u}^t + \nabla\Phi\left(\check{p}^{t+1}\right) - \nabla\Phi\left(p^t\right) = 0 \tag{4}$$

By Lemma C.1, the optimality condition for p^{t+1} implies that

$$\langle p - p^{t+1}, \nabla\Phi(p^{t+1}) - \nabla\Phi(\check{p}^{t+1}) \rangle \geq 0, \quad \text{for all } p \in \mathcal{A}.$$

Combining the last two displays, we have

$$\langle p - p^{t+1}, -\eta\hat{u}^t + \nabla\Phi\left(p^{t+1}\right) - \nabla\Phi\left(p^t\right) \rangle \geq 0, \quad \text{for all } p \in \mathcal{A}. \tag{5}$$

By Lemma C.1, this is the optimality condition for p^{t+1} to be the solution of (2). Note that (4) implies that

$$-\eta\hat{u}_i^t + \log\check{p}_i^{t+1} - \log p_i^t = 0 \quad \text{for all } i \in [n].$$

Therefore,

$$\check{p}_i^{t+1} = p_i^t \exp\left(\eta\hat{u}_i^t\right) = \tilde{p}_i^{t+1} \quad \text{for all } i \in [n].$$

and the final result follows from Lemma C.2.

$\square$

B.2. Proof of Theorem 4.1

Our proof is based on Lattimore & Szepesvári (2020, Chapter 31.1). Recall that we have

$$
\begin{aligned}
R_B &= \mathbb{E}\left[\max_{(a^t)\in\Gamma_m} K \sum_{t=0}^{T-1} \Delta U_{a^t}^t - \sum_{t=0}^{T} \Delta U_{\mathrm{cm}}^t \right] \\
&= \mathbb{E}\left[\max_{(a^t)\in\Gamma_m} \sum_{t=0}^{T-1}\sum_{k=0}^{K-1} \Delta U_{a^t}^t - \sum_{t=0}^{T}\sum_{i\in S^t} \Delta U_i^t \right] \\
&= \mathbb{E}\left[\max_{(a^t)\in\Gamma_m} \sum_{t=0}^{T-1}\sum_{k=0}^{K-1} \left(\Delta U_{a^t}^t - \Delta U_{i^{t,k}}^t \right) \right].
\end{aligned}
$$

Let

$$
\left(a_\star^0,\ldots,a_\star^{T-1}\right) = \operatorname{argmax}_{(a^t)\in\Gamma_m} \sum_{t=0}^{T-1} \Delta U_{a^t}^t,
$$

then there exists $0 = t_1 < t_2 < \ldots < t_m < t_{m+1} = T-1$ such that for any $j = 1,\ldots,m$, $a_\star^t$ is a constant when $t_j \le t < t_{j+1}$. To simplify the notation, we denote $\bar{a}_\star^j = a_\star^{t_j}$, we then decompose the regret as

$$
R_B = \mathbb{E}\left[\sum_{j=1}^{m}\sum_{t=t_j}^{t_{j+1}-1}\sum_{k=0}^{K-1} \left(\Delta U_{\bar{a}_\star^j}^t - \Delta U_{i^{t,k}}^t \right) \right] = \sum_{j=1}^{m}\mathbb{E}\left[\mathbb{E}\left[\sum_{t=t_j}^{t_{j+1}-1}\sum_{k=0}^{K-1} \left(\Delta U_{\bar{a}_\star^j}^t - \Delta U_{i^{t,k}}^t \right) \,\middle|\, p^{t_j} \right] \right].
$$

Let $u^t = (\Delta U_1^t,\ldots,\Delta U_n^t)^\top$. Note that since $U(\cdot)\in[0,1]$, we have $\Delta U_i^t \in [-1,1]$ for all $i\in[n]$. Besides, we have

$$
\begin{aligned}
&\mathbb{E}\left[\sum_{t=t_j}^{t_{j+1}-1}\sum_{k=0}^{K-1} \left(\Delta U_{\bar{a}_\star^j}^t - \Delta U_{i^{t,k}}^t \right) \,\middle|\, p^{t_j} \right] \\
&= \mathbb{E}\left[\max_{p\in\mathcal{P}_{n-1}} \sum_{t=t_j}^{t_{j+1}-1}\sum_{k=0}^{K-1} \langle p - p^t, u^t \rangle \,\middle|\, p^{t_j} \right] \\
&\le \mathbb{E}\left[\max_{p\in\mathcal{A}} \sum_{t=t_j}^{t_{j+1}-1}\sum_{k=0}^{K-1} \langle p - p^t, u^t \rangle \,\middle|\, p^{t_j} \right] + K\frac{\alpha}{n}(n-1)(t_{j+1}-t_j) \\
&= \mathbb{E}\left[\max_{p\in\mathcal{A}} K \sum_{t=t_j}^{t_{j+1}-1} \langle p - p^t, \hat{u}^t \rangle \,\middle|\, p^{t_j} \right] + K\alpha\left(1-\frac{1}{n}\right)(t_{j+1}-t_j). \quad (6)
\end{aligned}
$$

Since

$$
D_\Phi\left(x_1 \,\|\, x_2\right) = D_\Phi\left(x_3 \,\|\, x_2\right) + D_\Phi\left(x_1 \,\|\, x_3\right) + \langle \nabla\Phi(x_2) - \nabla\Phi(x_3), x_3 - x_1 \rangle, \qquad x_1, x_2, x_3 \in \mathcal{D}, \quad (7)
$$

then by (5), we have

$$
\begin{aligned}
\langle p - p^{t+1}, \hat{u}^t \rangle &\le \frac{1}{\eta}\langle p - p^{t+1}, \nabla\Phi\left(p^{t+1}\right) - \nabla\Phi\left(p^t\right)\rangle \\
&= \frac{1}{\eta}\left(D_\Phi\left(p \,\|\, p^t\right) - D_\Phi\left(p \,\|\, p^{t+1}\right) - D_\Phi\left(p^{t+1} \,\|\, p^t\right)\right).
\end{aligned}
$$

Thus, we have

$$
\sum_{t=t_j}^{t_{j+1}-1} \langle p - p^t, \hat{u}^t \rangle
$$

$$
\begin{aligned}
&= \sum_{t=t_j}^{t_{j+1}-1} \left\langle p^{t+1} - p^t, \hat{u}^t \right\rangle + \sum_{t=t_j}^{t_{j+1}-1} \left\langle p - p^{t+1}, \hat{u}^t \right\rangle \\
&\leq \sum_{t=t_j}^{t_{j+1}-1} \left\langle p^{t+1} - p^t, \hat{u}^t \right\rangle + \frac{1}{\eta} \sum_{t=t_j}^{t_{j+1}-1} D_\Phi \left(p \,\|\, p^t \right) - D_\Phi \left(p \,\|\, p^{t+1} \right) - D_\Phi \left(p^{t+1} \,\|\, p^t \right) \\
&= \sum_{t=t_j}^{t_{j+1}-1} \left\langle p^{t+1} - p^t, \hat{u}^t \right\rangle + \frac{1}{\eta} \left(D_\Phi \left(p \,\|\, p^{t_j} \right) - D_\Phi \left(p \,\|\, p^{t_{j+1}} \right) \right) - \frac{1}{\eta} \sum_{t=t_j}^{t_{j+1}-1} D_\Phi \left(p^{t+1} \,\|\, p^t \right) \\
&\leq \sum_{t=t_j}^{t_{j+1}-1} \left\langle p^{t+1} - p^t, \hat{u}^t \right\rangle + \frac{1}{\eta} D_\Phi \left(p \,\|\, p^{t_j} \right) - \frac{1}{\eta} \sum_{t=t_j}^{t_{j+1}-1} D_\Phi \left(p^{t+1} \,\|\, p^t \right) \tag{8}
\end{aligned}
$$

By (4), we have

$$
\hat{u}^t = \frac{1}{\eta} \left(\nabla \Phi \left(\tilde{p}^{t+1} \right) - \nabla \Phi \left(p^t \right) \right),
$$

and since (7), we have

$$
\begin{aligned}
\left\langle p^{t+1} - p^t, \hat{u}^t \right\rangle &= \frac{1}{\eta} \left\langle p^{t+1} - p^t, \nabla \Phi \left(\tilde{p}^{t+1} \right) - \nabla \Phi \left(p^t \right) \right\rangle \\
&= \frac{1}{\eta} \left(D_\Phi \left(p^t \,\|\, \tilde{p}^{t+1} \right) + D_\Phi \left(p^{t+1} \,\|\, p^t \right) - D_\Phi \left(p^{t+1} \,\|\, \tilde{p}^{t+1} \right) \right) \\
&\leq \frac{1}{\eta} \left(D_\Phi \left(p^t \,\|\, \tilde{p}^{t+1} \right) + D_\Phi \left(p^{t+1} \,\|\, p^t \right) \right). \tag{9}
\end{aligned}
$$

Combine (8) and (9), we have

$$
\sum_{t=t_j}^{t_{j+1}-1} \left\langle p - p^t, \hat{u}^t \right\rangle \leq \frac{1}{\eta} \left(D_\Phi \left(p \,\|\, p^{t_j} \right) + \sum_{t=t_j}^{t_{j+1}-1} D_\Phi \left(p^t \,\|\, \tilde{p}^{t+1} \right) \right). \tag{10}
$$

Note that

$$
D_\Phi \left(p^t \,\|\, \tilde{p}^{t+1} \right) = \sum_{i=1}^n p_i^t \left(\exp \left(\eta \hat{u}_i^t \right) - 1 - \eta \hat{u}_i^t \right) \leq \frac{\eta^2}{2} \sum_{i=1}^n p_i^t (\hat{u}_i^t)^2 \leq \frac{\eta^2 n}{2},
$$

thus we have

$$
\frac{1}{\eta} \sum_{t=t_j}^{t_{j+1}-1} D_\Phi \left(p^t \,\|\, \tilde{p}^{t+1} \right) \leq \frac{\eta n (t_{j+1} - t_j)}{2}. \tag{11}
$$

Combine (6), (10), and (11), we have

$$
\begin{aligned}
\mathbb{E} \left[\left. \sum_{t=t_j}^{t_{j+1}-1} \sum_{k=0}^{K-1} \left(\Delta U_{\bar{a}_\star^j}^{t,k} - \Delta U_{i^{t,k}}^{t,k} \right) \right| p^{t_j} \right] &\leq K\alpha \left(1 - \frac{1}{n} \right) (t_{j+1} - t_j) + \frac{\eta n K (t_{j+1} - t_j)}{2} \\
&\quad + K \mathbb{E} \left[\left. \max_{p \in \mathcal{A}} \frac{D_\Phi \left(p \,\|\, p^{t_j} \right)}{\eta} \right| p^{t_j} \right].
\end{aligned}
$$

Note that $p^{t_j} \in \mathcal{A}$, thus we have $p_i^{t_j} \geq \alpha/n$ for all $i \in [n]$ and

$$
D_\Phi \left(p \,\|\, p^{t_j} \right) = \sum_{i=1}^n p_i \log p_i + \sum_{i=1}^n p_i \log \left(\frac{1}{p_i^{t_j}} \right) \leq \log(n/\alpha),
$$

which implies that

$$
\mathbb{E} \left[\left. \sum_{t=t_j}^{t_{j+1}-1} \sum_{k=0}^{K-1} \left(\Delta U_{\bar{a}_\star^j}^{t,k} - \Delta U_{i^{t,k}}^{t,k} \right) \right| p^{t_j} \right] \leq K\alpha \left(1 - \frac{1}{n} \right) (t_{j+1} - t_j) + \frac{\eta n K (t_{j+1} - t_j)}{2} + \frac{K \log(n/\alpha)}{\eta},
$$

and

$$R_B \le \alpha KT + \frac{\eta n KT}{2} + \frac{mK \log(n/\alpha)}{\eta}.$$

The final result then follows by $B = KT$.

C. Useful lemmas

Lemma C.1. *Suppose that f is a differentiable convex function defined on domf, and $\mathcal{X} \subseteq$ domf is a closed convex set. Then x is the minimizer of f on $\mathcal{X}$ if and only if*

$$\nabla f(x)^\top (y - x) \ge 0 \quad \text{for all } y \in \mathcal{X}.$$

Proof. See Section 4.2.3 of Boyd et al. (2004). □

Lemma C.2 (Based on Exercise 26.12 of Lattimore & Szepesvári (2020) and Lemma 4 of Zhao et al. (2021)). *Let $\alpha \in [0, 1]$, $\mathcal{A} = \mathcal{P}_{n-1} \cap [\alpha/n, 1]^n$. For $y \in [0, \infty)^n$, let $x = \arg\min_{v \in \mathcal{A}} D_\Phi(v\|y)$. Suppose $y_1 \le y_2 \le \cdots \le y_n$. Let $i^\star$ be the smallest value such that*

$$y_{i^\star} \left(1 - \frac{i^\star - 1}{n}\alpha\right) > \frac{\alpha}{n} \sum_{i=i^\star}^n y_i.$$

Then

$$x_i = \begin{cases} \alpha/n & \text{if } i < i^\star \\ \dfrac{(1 - \frac{i^\star - 1}{n}\alpha)y_i}{\sum_{j=i^\star}^n y_j} & \text{otherwise.} \end{cases}$$

Proof. Consider the following constrained optimization problem:

$$\min_{u \in [0,\infty)^n} \sum_{i=1}^n u_i \log \frac{u_i}{y_i},$$

$$\text{s.t.} \sum_{i=1}^n u_i = 1,$$

$$u_i \ge \frac{\alpha}{n}, \qquad i \in [n].$$

Since x is the solution to this problem, by the optimality condition, there exists $\lambda, \nu_1, \ldots, \nu_n \in \mathbb{R}$ such that

$$\log \frac{x_i}{y_i} + 1 - \lambda - \nu_i = 0, \qquad i \in [n], \tag{12}$$

$$\sum_{i=1}^n x_i = 1, \tag{13}$$

$$x_i - \frac{\alpha}{n} \ge 0, \qquad i \in [n], \tag{14}$$

$$\nu_i \ge 0, \qquad i \in [n], \tag{15}$$

$$\nu_i \left(x_i - \frac{\alpha}{n}\right) = 0, \qquad i \in [n]. \tag{16}$$

By (12), we have $x_i = y_i \exp(-1 + \lambda + \nu_i)$. By (15) and (16), when $x_i = \alpha/n$, we have $x_i = y_i \exp(-1 + \lambda + \nu_i) \ge y_i \exp(-1 + \lambda)$; when $x_i > \alpha/n$, we have $x_i = y_i \exp(-1 + \lambda)$. Assume that $x_1 = \cdots = x_{i^\star - 1} = \alpha/n < x_{i^\star} \le \cdots \le x_n$. Then

$$1 = \sum_{i=1}^n x_i = (i^\star - 1)\frac{\alpha}{n} + \exp(-1 + \lambda) \cdot \sum_{i=i^\star}^n y_i,$$

which implies that

$$\exp(-1 + \lambda) = \frac{1 - (i^\star - 1)\frac{\alpha}{n}}{\sum_{i=i^\star}^n y_i}. \tag{17}$$

16

Thus, we have

$$x_{i^\star} = y_{i^\star} \exp(-1 + \lambda) = y_{i^\star} \frac{1 - (i^\star - 1)\frac{\alpha}{n}}{\sum_{i=i^\star}^n y_i} > \frac{\alpha}{n},$$

which implies that

$$y_{i^\star} \left(1 - \frac{i^\star - 1}{n} \alpha \right) > \frac{\alpha}{n} \sum_{i=i^\star}^n y_i. \tag{18}$$

To complete the proof, we then only need to show that

$$y_{i'} \left(1 - \frac{i' - 1}{n} \alpha \right) \leq \frac{\alpha}{n} \sum_{i=i'}^n y_i \tag{19}$$

for all $1 \leq i' \leq i^\star - 1$. The result then follows from (18) and (19). To prove(19), recall that for any $1 \leq i' \leq i^\star - 1$, we have $\alpha/n = y_{i'} \exp(-1 + \lambda + \nu_{i'})$, and because $y_1 \leq \cdots \leq y_n$, we have $\nu_1 \geq \cdots \geq \nu_{i^\star - 1}$. This way, we have

$$(i^\star - i') \frac{\alpha}{n} = \sum_{i=i'}^{i^\star - 1} y_i \exp(-1 + \lambda + \nu_i) \leq \exp(-1 + \lambda + \nu_{i'}) \sum_{i=i'}^{i^\star - 1} y_i. \tag{20}$$

On the other hand, by (17), we have

$$1 - (i^\star - 1)\frac{\alpha}{n} = \exp(-1 + \lambda) \sum_{i=i^\star}^n y_i \leq \exp(-1 + \lambda + \nu_{i'}) \sum_{i=i^\star}^n y_i. \tag{21}$$

Combining (20) and (21), we have

$$
\begin{aligned}
\frac{1 - (i' - 1)\frac{\alpha}{n}}{\sum_{i=i'}^n y_i} &= \frac{1 - (i^\star - 1)\frac{\alpha}{n} + (i^\star - i')\frac{\alpha}{n}}{\sum_{i=i'}^n y_i} \\
&\leq \frac{\exp(-1 + \lambda + \nu_{i'}) \left(\sum_{i=i'}^{i^\star - 1} y_i + \sum_{i=i^\star}^n y_i \right)}{\sum_{i=i'}^n y_i} \\
&= \exp(-1 + \lambda + \nu_{i'}) \\
&= \frac{\frac{\alpha}{n}}{y_{i'}},
\end{aligned}
$$

which then implies (19). $\qquad\square$