Shapley Value Estimation Based on Differential Matrix

JUNYUAN PANG, Zhejiang University, China
JIAN PEI, Duke University, USA
HAOCHENG XIA, University of Illinois Urbana-Champaign, USA
XIANG LI, Zhejiang University, China
JINFEI LIU*, Zhejiang University, China

The Shapley value has been extensively used in many fields as the unique metric to fairly evaluate player contributions in cooperative settings. Since the exact computation of Shapley values is #P-hard in the task-agnostic setting, many studies have been developed to utilize the Monte Carlo method for Shapley value estimation. The existing methods estimate the Shapley values directly. In this paper, we explore a novel idea—inferring the Shapley values by estimating the differences between them. Technically, we estimate a differential matrix consisting of pairwise Shapley value differences to reduce the variance of the estimated Shapley values. We develop a least-squares optimization solution to derive the Shapley values from the differential matrix, minimizing the estimator variances. Additionally, we devise a Monte Carlo method for efficient estimation of the differential matrix and introduce two stratified Monte Carlo methods for further variance reduction. Our experimental results on real and synthetic data sets demonstrate the effectiveness and efficiency of the differential-matrix-based sampling approaches.

CCS Concepts: • **Information systems → Data management systems**;

Additional Key Words and Phrases: Shapley value; Sampling

ACM Reference Format:
Junyuan Pang, Jian Pei, Haocheng Xia, Xiang Li, and Jinfei Liu. 2025. Shapley Value Estimation Based on Differential Matrix. *Proc. ACM Manag. Data* 3, 1 (SIGMOD), Article 75 (February 2025), 28 pages. https://doi.org/10.1145/3709725

1 Introduction

The renowned Shapley value [60] is the unique metric for fair reward allocation towards a collective utility among contributors in a cooperative game. It is grounded in four fundamental desiderata of fairness, namely efficiency, symmetry, dummy player, and additivity. Owing to the flexibility of the utility functions, the Shapley value demonstrates significant universality and adaptability across various areas in data management, such as data pricing [1, 10, 20, 41, 42, 46, 47], data/feature selection [19, 24], and explanations in database queries [2, 6, 14, 16, 31, 43].

Despite its widespread applicability in numerous domains, applications of the Shapley value face considerable computational challenges. Specifically, the Shapley value for a player z measures the weighted average of the player's *marginal contributions* $\mathcal{U}(\mathcal{S} \cup \{z\}) - \mathcal{U}(\mathcal{S})$ $(z \notin \mathcal{S})$ for all

*Corresponding author.

Authors' Contact Information: Junyuan Pang, Zhejiang University, China, junyuanpang@zju.edu.cn; Jian Pei, Duke University, USA, j.pei@duke.edu; Haocheng Xia, University of Illinois Urbana-Champaign, USA, hxia7@illinois.edu; Xiang Li, Zhejiang University, China, lixiangzx@zju.edu.cn; Jinfei Liu, Zhejiang University, China, jinfeiliu@zju.edu.cn.

ACM 2836-6573/2025/2-ART75
https://doi.org/10.1145/3709725

2^{n-1} possible coalitions, where $\mathcal{S}$ is a coalition of players, $\mathcal{U}(\cdot)$ is a utility function, and n is the number of players. The need to evaluate utilities for an exponential number of coalitions makes the exact computation of Shapley values #P-hard [15] in the task-agnostic setting. For large-scale applications, particularly database fact valuation in query answering [2, 6, 14, 16, 31, 43] and data valuation in machine learning [19, 24], the #P-hardness computational complexity renders the utilization of exact Shapley values impractical.

To facilitate the applications of the Shapley value to large-scale scenarios, approximation methods based on sampling have been extensively explored [7, 8, 30, 34, 44, 49–51, 72]. The Monte Carlo-based approximation algorithm [50] naturally considers the Shapley value as the weighted expectation of the marginal contributions of each player and samples the marginal contributions. The idea of sampling inspires a series of studies [7, 8, 30, 34, 37, 49, 51, 72]. Those existing methods estimate the Shapley values of different parties independently and separately.

Different from all the existing studies, in this paper we take advantage of the efficiency [60], a key property of the Shapley values. The property states that the Shapley values of all players sum to a constant, that is, $\sum_{z\in\mathcal{N}} \mathcal{SV}(z) = \mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset)$. This property allows for paired sampling and estimation, potentially reducing estimation variance. Despite its potential, surprisingly this idea has not been explored in the literature. The principled idea motivates our study. To illustrate the benefit of exploiting efficiency, we present the following example.

Example 1.1. Consider three i.i.d. random variables $X, Y, Z \sim \mathcal{N}(\frac{1}{3}, \sigma^2)$ such that $\mathrm{E}[X] + \mathrm{E}[Y] + \mathrm{E}[Z] = 1$. We want to estimate $x = \mathrm{E}[X]$, $y = \mathrm{E}[Y]$, and $z = \mathrm{E}[Z]$. Let us examine how the constraint $\mathrm{E}[X] + \mathrm{E}[Y] + \mathrm{E}[Z] = 1$ may help in the estimation.

First, as the *baseline method*, let us ignore the constraint and tackle the three independent random variables $\hat{x} \sim P(X)$, $\hat{y} \sim P(Y)$, and $\hat{z} \sim P(Z)$. Apparently, $\mathrm{Var}(\hat{x}) = \mathrm{Var}(\hat{y}) = \mathrm{Var}(\hat{z}) = \sigma^2$, and $\mathrm{Var}(\hat{x}) + \mathrm{Var}(\hat{y}) + \mathrm{Var}(\hat{z}) = \mathrm{Var}(X) + \mathrm{Var}(Y) + \mathrm{Var}(Z) = 3\sigma^2$.

Alternatively, as the *constraint-aware method* we consider the constraint and tackle the three independent random variables $\hat{a} \sim P(X - Y)$, $\hat{b} \sim P(Y - Z)$, and $\hat{c} \sim P(Z - X)$. We consider three random variables $\hat{x}' = \frac{1}{3} + \frac{\hat{a}}{3} - \frac{\hat{c}}{3}$, $\hat{y}' = \frac{1}{3} + \frac{\hat{b}}{3} - \frac{\hat{a}}{3}$, and $\hat{z}' = \frac{1}{3} + \frac{\hat{c}}{3} - \frac{\hat{b}}{3}$. Due to the constraint $x+y+z = 1$, we have $\mathrm{E}[\hat{x}'] = x = \mathrm{E}[X]$, $\mathrm{E}[\hat{y}'] = y = \mathrm{E}[Y]$, and $\mathrm{E}[\hat{z}'] = z = \mathrm{E}[Z]$. In other words, by estimating the expectations of $\hat{x}'$, $\hat{y}'$, and $\hat{z}'$ we can approach the expectations of X, Y, and Z.

Note that $X - Y, Y - Z, Z - X \sim \mathcal{N}(0, 2\sigma^2)$. We have $\mathrm{Var}(\hat{a}) = \mathrm{Var}(\hat{b}) = \mathrm{Var}(\hat{c}) = 2\sigma^2$. Then, in this constraint-aware method, $\mathrm{Var}(\hat{x}') = \frac{1}{9}(\mathrm{Var}(\hat{a}) + \mathrm{Var}(\hat{c})) = \frac{4}{9}\sigma^2$. Similarly, $\mathrm{Var}(\hat{y}') = \mathrm{Var}(\hat{z}') = \frac{4}{9}\sigma^2$. The variances of $\hat{x}', \hat{y}', \hat{z}'$ in this constraint-aware method are substantially smaller than the variances of $\hat{x}, \hat{y}, \hat{z}$ in the baseline approach.

Furthermore, we can consider the dependence of X, Y, Z and $\hat{a}, \hat{b}, \hat{c}$. Then, $\mathrm{Var}(\hat{x}') + \mathrm{Var}(\hat{y}') + \mathrm{Var}(\hat{z}') = \frac{2}{9}(\mathrm{Var}(\hat{a}) + \mathrm{Var}(\hat{b}) + \mathrm{Var}(\hat{c}) - \mathrm{Cov}(\hat{a}, \hat{b}) - \mathrm{Cov}(\hat{a}, \hat{c}) - \mathrm{Cov}(\hat{b}, \hat{c}))$. Since $\mathrm{Var}(\hat{a} + \hat{b} + \hat{c}) = \mathrm{Var}(\hat{a}) + \mathrm{Var}(\hat{b}) + \mathrm{Var}(\hat{c}) + 2\mathrm{Cov}(\hat{a}, \hat{b}) + 2\mathrm{Cov}(\hat{a}, \hat{c}) + 2\mathrm{Cov}(\hat{b}, \hat{c}) \geq 0$, we have $-\mathrm{Cov}(\hat{a}, \hat{b}) - \mathrm{Cov}(\hat{a}, \hat{c}) - \mathrm{Cov}(\hat{b}, \hat{c}) \leq \frac{1}{2}(\mathrm{Var}(\hat{a}) + \mathrm{Var}(\hat{b}) + \mathrm{Var}(\hat{c}))$. Therefore, $\mathrm{Var}(\hat{x}') + \mathrm{Var}(\hat{y}') + \mathrm{Var}(\hat{z}') \leq \frac{1}{3}(\mathrm{Var}(\hat{a}) + \mathrm{Var}(\hat{b}) + \mathrm{Var}(\hat{c}))$. Apply the same approach to $\mathrm{Var}(\hat{a}) + \mathrm{Var}(\hat{b}) + \mathrm{Var}(\hat{c})$, we have $\mathrm{Var}(\hat{a}) + \mathrm{Var}(\hat{b}) + \mathrm{Var}(\hat{c}) \leq 3(\mathrm{Var}(X) + \mathrm{Var}(Y) + \mathrm{Var}(Z))$. Therefore, $\mathrm{Var}(\hat{x}') + \mathrm{Var}(\hat{y}') + \mathrm{Var}(\hat{z}') \leq \mathrm{Var}(X) + \mathrm{Var}(Y) + \mathrm{Var}(Z) = 3\sigma^2$. □

Specifically, we can derive the Shapley values from their differences given the sum of the Shapley values, which can achieve lower variances than estimating the Shapley values independently and directly (see details in Section 4.1). Inspired by this intuition, we introduce *differential matrix*, an innovative concept representing all pairwise differences between the Shapley values, leading to more accurate estimates. We subsequently employ a least-squares optimization approach that fully leverages the differential matrix to determine the Shapley values (Theorem 4.3).

To efficiently approximate the differential matrix, we develop a Monte Carlo method by reformulating the pairwise difference between the Shapley values (Theorem 5.1). For further variance reduction, we develop a stratified Monte Carlo method based on the coalition size for the differential matrix estimation. Moreover, we calculate the optimal sample allocation to minimize the variances of the elements in the estimated differential matrix (Theorem 5.8). Due to the challenge of the unobservable stratum variances, we conduct preliminary sampling to estimate the variance of each stratum. Based on the stratum variances, we calculate an approximately optimal sample allocation. The superiority of the proposed method is demonstrated through mathematical formalizations (Theorems 6.1, 6.3, and 6.4) and empirical results (Section 7).

The Shapley value is of vital importance in various fields, which has spurred a large body of studies dedicated to computation given the formidable computational complexity. The primary novelty of this paper lies in investigating the potential of the differential matrix in Shapley value estimation. This innovation improves the efficiency and effectiveness of Shapley value estimation. Our main contributions are summarized as follows.

- We develop the difference matrix, an innovative concept representing all pairwise differences between Shapley values, to estimate the Shapley values more efficiently with the constraint of the efficiency property.
- We propose a suite of advanced sampling algorithms designed to efficiently estimate the difference matrix, leveraging both unstratified and stratified sampling techniques to enhance computational performance.
- Through mathematical analysis, we establish the theoretical superiority of our algorithms.
- Experiments are conducted on cooperative games and data valuation tasks to verify the efficiency and effectiveness of the proposed algorithms.

The rest of this paper is organized as follows. Section 2 provides a brief review of the existing research on the Shapley value and its computation and estimation methods. Section 3 revisits the concept of the Shapley value and various methods for the Shapley value estimation. Section 4 introduces the differential matrix, accompanied by a novel method to compute the Shapley values based on the differential matrix. Section 5 proposes several efficient algorithms for differential matrix estimation. Section 6 offers a theoretical analysis of the superiority of using the differential matrix. Experimental results and associated findings are presented in Section 7. The paper concludes in Section 8, summarizing our key insights and contributions. To keep the main body of the paper concise, we move all proofs to the appendix.

2 Related Work

The Shapley value [60], named in honor of Lloyd Shapley, plays a pivotal role in cooperative game theory and finds diverse applications in data management, including explanations in query answers [2, 4, 6, 14, 16, 31, 33, 43, 57], database tunning [35, 74], data debugging [32, 38, 59], data/feature selection [12, 23, 23, 24, 56, 66], data cleaning [19, 25], data pricing [1, 3, 9, 10, 20, 41, 42, 45–47, 71], model interpretation [26, 36, 44, 52, 56], client evaluation in federated learning [11, 17, 18, 48, 61, 62, 64, 65, 75], architecture search for graph neutral network [76], and influence maximization in the social network [5, 22, 77].

In the context of query answering, Livshits et al. [43] applied Shapley values to quantify the contribution of tuples to query results, specifically for conjunctive and aggregate queries, and proposed approximation algorithms to address computational challenges in complex cases. Reshef et al. [57] further investigated the computational complexity of determining Shapley values for conjunctive queries involving negation, highlighting additional hurdles in such scenarios. Building on this, Deutch et al. [16] leveraged the Shapley value as an explanation mechanism in databases by

assigning importance scores to facts. Moreover, Bienvenu et al. [6] linked Shapley value computation to fixed-size generalized model counting, offering insights into the complexity landscape for various types of queries. Complementing these studies, Karmakar et al. [33] explored the efficient computation of expected Shapley-like values within probabilistic databases, designing algorithms to streamline this process.

In the context of data valuation, which encompasses data selection, data pricing, and data cleaning, Ghorbani and Zou [24] pioneered the application of the Shapley value to quantify the value of data points in terms of their contribution to model performance. For data selection, Xia et al. [70] argued that probability, rather than accuracy, is a more appropriate utility function for Shapley value-based data valuation. In the domain of data cleaning, Farchi et al. [19] proposed a method to identify misclassified data points and rank data slices using Shapley values. For data pricing, Liu et al. [42] utilized Shapley values to allocate compensation in an end-to-end data marketplace.

Since the computation of Shapley values is proved to be #P-hard [15], various approximation techniques have been developed [7, 13, 24, 28–30, 34, 37, 44, 49–51, 63, 67, 69, 72, 73]. Mann and Shapley [50] first applied the Monte Carlo sampling technique on Shapley value estimation. Subsequent research includes permutation sampling methods that estimate Shapley values as expectations of marginal contributions [51], enhanced by Quasi-Monte Carlo techniques [49] and stratified sampling algorithms [28] as improved by Castro et al. [7]. Lundberg and Lee [44] computed the Shapley value by solving a weighted optimization based on sampled utilities, improved by Covert and Lee [13] based on paired sampling. As computing Shapley values in general cases is particularly challenging, Jia et al. [29] devised algorithms with polynomial complexity specifically for KNN classifiers. Besides, Ghorbani and Zou [24] omitted marginal contributions below a threshold or approximated utilities based on gradients in permutation sampling, enhancing efficiency at the expense of the unbiased guarantee. In addition to computing Shapley values within a fixed dataset, Zhang et al. [73] explored efficient approximation algorithms for updating Shapley values for dynamic datasets.

The most related works are [30, 37, 72]. Zhang et al. [72] introduced the concept of complementary contribution to replace marginal contributions, opening avenues for different sample forms beyond traditional marginal contributions. Each sample can be used to estimate the Shapley values of all players. Li and Yu [37] transformed the Shapley value into a form of utility combinations by adding a constant to each Shapley value, reusing the utility with an expectation of $n/2$ times in Shapley value estimation. Moreover, Kolpaczki et al. [34] proposed a very similar sampling technique using utilities by splitting the Shapley value into two terms. Jia et al. [30] devised sampling algorithms based on group testing, which also reuses utilities by sampling differences between Shapley values. These methods collectively address the computational efficiency bottleneck in Shapley value estimation by emphasizing utility reuse or alternative sampling strategies. However, our work differs fundamentally in its approach. Instead of directly estimating the Shapley values or relying solely on utility reuse, we focus on leveraging the efficiency axiom and paired sampling to estimate Shapley values based on the differential matrix. This shift in perspective enables further improvements in computational efficiency and accuracy, setting our method apart from existing approaches.

3 Preliminaries

In this section, we revisit the concept of the Shapley value and review three methods for Shapley value estimation, including the classical method based on marginal contributions [8] and two state-of-the-art methods without using marginal contributions [37, 72]. Table 1 summarizes the frequently used notations.

Table 1. A summary of frequently used notations.

Notation	Definition
n	the number of players
m	the number of samples
$\mathcal{U}(\cdot)$	utility function
$\mathcal{N}$	a set of players
z_i	the i^{th} player
$\mathcal{S}$	a coalition within $\mathcal{N}$
$\mathcal{SV}_i$	the Shapley value of z_i
$\Delta\mathcal{SV}_{i,j}$	the difference between $\mathcal{SV}_i$ and $\mathcal{SV}_j$
$\widehat{\mathcal{SV}_i}$	the estimated Shapley value of z_i
$\widehat{\Delta\mathcal{SV}_{i,j}}$	the estimator of $\Delta\mathcal{SV}_{i,j}$

3.1 The Shapley Value and MC Estimation

Consider a set $\mathcal{N} = \{z_1, \ldots, z_n\}$ of n players. A **coalition** $\mathcal{S} \subseteq \mathcal{N}$ is a group of players collaborating to fulfill a task. $\mathcal{N}$ itself as a coalition is called the **grand coalition**. A utility function $\mathcal{U}(\mathcal{S})$ measures the utility of coalition $\mathcal{S}$ for a task. The **marginal contribution** of a player z_i ($1 \le i \le n$) to a coalition $\mathcal{S}$ is given by $\mathcal{U}(\mathcal{S} \cup \{z_i\}) - \mathcal{U}(\mathcal{S})$.

The **Shapley value** [60] $\mathcal{SV}_i$ of a player z_i is the weighted expectation of the marginal contributions by z_i to all possible coalitions over $\mathcal{N}$, that is

$$\mathcal{SV}_i = \frac{1}{n} \sum_{\mathcal{S} \subseteq \mathcal{N} \setminus \{z_i\}} \frac{\mathcal{U}(\mathcal{S} \cup \{z_i\}) - \mathcal{U}(\mathcal{S})}{\binom{n-1}{|\mathcal{S}|}} \tag{1}$$

$$= \frac{1}{n!} \sum_{\pi \in \Pi(\mathcal{N})} (\mathcal{U}(P^{\pi}_{z_i} \cup \{z_i\}) - \mathcal{U}(P^{\pi}_{z_i})), \tag{2}$$

where $\Pi(\mathcal{N})$ is the set of all permutations on $\mathcal{N}$ and $P^{\pi}_{z_i}$ is the set of players preceding z_i in permutation π.

The Shapley value establishes the foundational criteria for fair reward allocation, encompassing *efficiency*, *symmetry*, *dummy player*, and *additivity* [58]. Specifically, **efficiency** requires that the utility of the grand coalition should be completely distributed among all participants, that is, $\sum_{i=1}^{n} \mathcal{SV}_i = \mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset)$. **Symmetry** dictates that two players always with equivalent marginal contributions should receive identical Shapley values. **Dummy player** indicates that a player always with zero marginal contributions in all cases should have the Shapley value of 0. **Additivity** states that the Shapley value of a player in one game that is the union of two tasks should be the sum of the Shapley values in those individual games.

Computing the exact Shapley value using Equation 1 directly requires enumerating all possible subsets of $\mathcal{N}$, which is computationally prohibitive for a large number of players. Moreover, utility evaluation may be costly in some applications, such as constructing an advanced machine-learning model and evaluating the model performance.

The Monte Carlo method [8] is widely adopted to estimate the Shapley values using Equation 2. It randomly samples permutations of all players and computes the marginal contribution of each player z_i in the sampled permutations. Subsequently, the Shapley value of player z_i is estimated as the average of the marginal contributions for player z_i in all sampled permutations. The Monte Carlo method provides an unbiased estimator of the Shapley value and achieves better accuracy with more samples.

However, the traditional Monte Carlo methods based on marginal contributions may still be costly since one marginal contribution $\mathcal{U}(\mathcal{S} \cup \{z_i\}) - \mathcal{U}(\mathcal{S})$ can only be used to estimate the Shapley value of one player z_i. This limitation is particularly evident in situations where the Shapley values of multiple players or even all players are computed in a shot, where one has to evaluate the utilities of many coalitions and utility evaluation is costly.

3.2 Monte Carlo Methods without Marginal Contributions

Recently, in order to improve the utility of the evaluation of utility functions, some Monte Carlo methods [37, 72] for Shapley value estimation not based on marginal contributions are proposed.

3.2.1 Estimation based on complementary contributions. Zhang et al. [72] make an insightful observation about the **complementary contribution** of a coalition $\mathcal{S}$, defined by $CC_{\mathcal{N}}(\mathcal{S}) = \mathcal{U}(\mathcal{S}) - \mathcal{U}(\mathcal{N}\backslash\mathcal{S})$. The utilities $\mathcal{U}(\mathcal{S})$ and $\mathcal{U}(\mathcal{N} \setminus \mathcal{S})$ have total weights $\binom{n-1}{|\mathcal{S}|-1}$ and $-\binom{n-1}{n-|\mathcal{S}|}$, respectively, in computing Shapley value $\mathcal{SV}_i$ ($z_i \in \mathcal{S}$) using Equation 1. Therefore, the Shapley value can be estimated using complementary contributions, that is, given a set of players $\mathcal{N}$, the Shapley value of z_i is

$$\mathcal{SV}_i = \frac{1}{n} \sum_{\mathcal{S} \subseteq \mathcal{N}\setminus\{z_i\}} \frac{CC_{\mathcal{N}}(\mathcal{S} \cup \{z_i\})}{\binom{n-1}{|\mathcal{S}|}}.$$

For any coalition $\mathcal{S} \subseteq \mathcal{N}$, since every data player z_i appears in either $\mathcal{S}$ or its complement coalition $\mathcal{N} \setminus \mathcal{S}$, the estimation of the Shapley value of every player z_i can use either $\mathcal{U}(\mathcal{S})$ or $\mathcal{U}(\mathcal{N}\setminus\mathcal{S})$. In other words, for a sampled coalition $\mathcal{S}$, the calculation of utilities $\mathcal{U}(\mathcal{S})$ and $\mathcal{U}(\mathcal{N}\setminus\mathcal{S})$ can be used in the Shapley value estimation of every data player, and thus the utilization of the samples and the utility evaluation can be improved substantially.

3.2.2 Estimation based on utilities. Li and Yu [37] make another insightful observation about the Shapley value computation by adding a dummy player z_0 and rewriting Equation 1 into

$$\mathcal{SV}_i = \frac{1}{n} \sum_{\mathcal{S} \subseteq \mathcal{N}\cup\{z_0\}\setminus\{z_i\}} \frac{\mathcal{U}(\mathcal{S} \cup \{z_i\})}{\binom{n-1}{|\mathcal{S}|}} - \frac{1}{n} \sum_{\mathcal{S} \subseteq \mathcal{N}} \frac{\mathcal{U}(\mathcal{S} \cup \{z_0\})}{\binom{n-1}{|\mathcal{S}|}}. \tag{3}$$

Since the second term in Equation 3 is a constant, we only need to estimate the first term $\frac{1}{n}\sum_{\mathcal{S}\subseteq\mathcal{N}\cup\{z_0\}\setminus\{z_i\}} \frac{\mathcal{U}(\mathcal{S}\cup\{z_i\})}{\binom{n-1}{|\mathcal{S}|}}$ ($0 \le i \le n$). For any coalition $\mathcal{S} \subseteq \mathcal{N} \cup \{z_0\}$, the corresponding utility $\mathcal{U}(\mathcal{S})$ calculated can be used in the estimation of the Shapley value of every player $z_i \in \mathcal{S}$, and thus the utilization of the sample coalitions and the utility evaluation can also be improved substantially.

4 Shapley value with Differential Matrix

In this section, we develop a new method of differential matrix to reconstruct original values from differences. We first demonstrate our mathematical intuition of differential matrix in Section 4.1. We then introduce the concept of differential matrix and identify some useful properties in Section 4.2. Finally, we present our new approach to determining Shapley values based on the differential matrix by solving a least-squares optimization in Section 4.3.

4.1 Intuition

In the existing methods, the Shapley value of each player is approximated separately. However, the Shapley values of all players have the nice property of efficiency, that is $\sum_{i=1}^{n} \mathcal{SV}_i = \mathcal{U}(\mathcal{N})$. Can this efficiency property help us estimate Shapley values faster and more accurately?

Example 4.1 (Intuition). Consider n i.i.d random variables $X_1, X_2, \ldots, X_n \sim \mathcal{N}(\frac{1}{n}, \sigma^2)$ such that $\sum_{i=1}^{n} \mathrm{E}[X_i] = 1$. We want to estimate $x_i = \mathrm{E}[X_i] (i = 1, 2, \ldots, n)$. There are also two approaches for the estimation.

First, let us ignore the constraint and tackle the independent random variables $\hat{x}_i \sim P(X_i) (i = 1, 2, \ldots, n)$. Apparently, $\mathrm{Var}(\hat{x}_1) = \mathrm{Var}(\hat{x}_2) = \cdots = \mathrm{Var}(\hat{x}_n) = \sigma^2$, and $\mathrm{Var}(\hat{x}_1) + \mathrm{Var}(\hat{x}_2) + \cdots + \mathrm{Var}(\hat{x}_n) = n\sigma^2$.

Alternatively, we can consider the constraint and tackle the independent random variables $\hat{d}_{i,j} \sim P(X_i - X_j) (1 \leq i < j \leq n)$. We consider the random variables $\hat{x}'_i = \frac{1}{n} + \frac{1}{n} \sum_{j=1}^{n} \hat{d}_{i,j}$. Due to the constraint $\sum_{i=1}^{n} x_i = 1$, we have $\mathrm{E}[\hat{x}'_i] = x_i = \mathrm{E}[X_i] (i = 1, 2, \ldots, n)$.

Note that $X_i - X_j \sim \mathcal{N}(0, 2\sigma^2) (1 \leq i < j \leq n)$. We have $\mathrm{Var}(d_{i,j}) = 2\sigma^2$. Then, in this alternative method, $\mathrm{Var}(\hat{x}'_i) = \frac{1}{n^2} \sum_{j \neq i} \mathrm{Var}(d_{i,j}) = \frac{2(n-1)}{n^2} (i = 1, 2, \ldots, n)$. The variances of $\hat{x}'_i$ in this alternative method are smaller than the variances of $\hat{x}_i$ in the first approach.

The above example clearly shows that constraints among the expectations of random variables may help us obtain more accurate estimations of the expectations of random variables. As the property of efficiency is such a constraint, the intuition of our approach is to make good use of the property in the Shapley value estimation.

4.2 Differential Matrix

Motivated by Example 4.1, we introduce the notion of differential matrix.

Definition 4.2 (Differential matrix). Given a set of players $\mathcal{N} = \{z_1, \ldots, z_n\}$, the **differential matrix of Shapley values** $\Delta\mathcal{SV}$ is an $n \times n$ matrix comprising all the pairwise differences of the Shapley values between players, where element $\Delta\mathcal{SV}_{i,j}$ in the i^{th} row and the j^{th} column is $\Delta\mathcal{SV}_{i,j} = \mathcal{SV}_i - \mathcal{SV}_j$, the difference between the Shapley values of z_i and z_j. □

A differential matrix has the following useful properties.

Property 1. *Given a set of players $\mathcal{N} = \{z_1, \ldots, z_n\}$, the differential matrix of Shapley values $\Delta\mathcal{SV}$ has the following properties.*

- *Anti-symmetricity:* $\forall 1 \leq i, j \leq n, \Delta\mathcal{SV}_{i,j} = -\Delta\mathcal{SV}_{j,i}$;
- *Zero diagonal:* $\forall 1 \leq i \leq n, \Delta\mathcal{SV}_{i,i} = 0$; *and*
- *Triangularity:* $\forall 1 \leq i, j, l \leq n, \Delta\mathcal{SV}_{i,j} = \Delta\mathcal{SV}_{l,j} - \Delta\mathcal{SV}_{l,i}$. □

The differential matrix is a skew-symmetric matrix according to the anti-symmetricity and zero diagonal and thus can be transformed into an upper-triangular matrix containing $n(n-1)/2$ elements as follows.

$$\begin{pmatrix} 0 & \Delta\mathcal{SV}_{1,2} & \Delta\mathcal{SV}_{1,3} & \cdots & \Delta\mathcal{SV}_{1,n} \\ & 0 & \Delta\mathcal{SV}_{2,3} & \cdots & \Delta\mathcal{SV}_{2,n} \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & \Delta\mathcal{SV}_{n-1,n} \\ & & & & 0 \end{pmatrix}$$

4.3 From Differential Matrix to Shapley Values

How can we compute the Shapley values using the differential matrix?

Using the efficiency axiom of the Shapley value $\sum_{i=1}^{n} \mathcal{SV}_i = \mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset)$, we can obtain the Shapley values of all players by solving the following system of linear equations.

$$\begin{cases} n\mathcal{SV}_1 - \sum_{j=1}^{n} \Delta\mathcal{SV}_{1,j} = \mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset), \\ \vdots \\ n\mathcal{SV}_n - \sum_{j=1}^{n} \Delta\mathcal{SV}_{n,j} = \mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset). \end{cases} \tag{4}$$

Alternatively, using least-squares optimization, we can derive the closed-form expression of the Shapley value.

Theorem 4.3. *The Shapley values $\mathcal{SV}_i$ $(1 \le i \le n)$ are the solutions to the following least-squares optimization problem.*

$$\min_{(\mathcal{SV}_1,\ldots,\mathcal{SV}_n)^T \in \mathbb{R}^n} \sum_{1 \le i < j \le n} |\mathcal{SV}_i - \mathcal{SV}_j - \Delta\mathcal{SV}_{i,j}|^2,$$

$$s.t. \quad \sum_{i=1}^{n} \mathcal{SV}_i = \mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset).$$

The closed-form expression of $\mathcal{SV}_i$ is

$$\mathcal{SV}_i = \frac{1}{n} \sum_{j=1}^{n} \Delta\mathcal{SV}_{i,j} + \frac{1}{n}[\mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset)], \tag{5}$$

which is also the solution to Equation 4. □

The Shapley value of player z_i obtained with Theorem 4.3 can be interpreted as the combination of the average difference compared to the other players and a uniform utility allocation.

5 Differential Matrix Estimation

Based on Theorem 4.3, we can approximate Shapley values from an estimated differential matrix. Now, the only question remained is how we can estimate the differential matrix effectively.

In this section, we systematically propose a series of techniques for estimating the differential matrix. We start with a simple Monte Carlo estimation approach using utilities in Section 5.1. Then, we explore the idea of approximating the differential matrix using stratified sampling and investigate two methods of sample allocation in Sections 5.2 and 5.3, respectively.

5.1 Differential Matrix Estimation Using Utilities

We can estimate the differential matrix directly by approximating each element in the matrix. We have the following result.

Theorem 5.1. *Given a set of players $\mathcal{N}$ and two players $z_i, z_j \in \mathcal{N}$, the difference between the Shapley values of z_i and z_j is*

$$\Delta\mathcal{SV}_{i,j} = \frac{1}{n-1} \sum_{\mathcal{S} \subseteq \mathcal{N} \setminus \{z_i, z_j\}} \frac{\mathcal{U}(\mathcal{S} \cup \{z_i\}) - \mathcal{U}(\mathcal{S} \cup \{z_j\})}{\binom{n-2}{|\mathcal{S}|}} \tag{6}$$

$$= \frac{1}{n-1} \sum_{\mathcal{S} \subseteq \mathcal{N} \setminus \{z_i, z_j\}} \left(\frac{\mathcal{U}(\mathcal{S} \cup \{z_i\})}{\binom{n-2}{|\mathcal{S}|}} - \frac{\mathcal{U}(\mathcal{S} \cup \{z_j\})}{\binom{n-2}{|\mathcal{S}|}} \right). \quad \square \tag{7}$$

Algorithm 1: Differential matrix estimation using utilities

input : a set of players $\mathcal{N} = \{z_1, \ldots, z_n\}$ and sample size $m > 0$
output: the estimated difference $\widehat{\Delta\mathcal{SV}_{i,j}}$

$\widehat{\mathcal{U}_{i,j}}, m_{i,j} \leftarrow 0\ (1 \le i, j \le n)$;
for $t = 1$ *to* m **do**
  Select k with probability $p_k \propto \frac{1}{k(n-k)}\ (1 \le k \le n-1)$;
  Let π^t be a random permutation of $\{1, \ldots, n\}$;
  $\mathcal{S} \leftarrow \{z_{\pi^t(1)}, \ldots, z_{\pi^t(k)}\}$;
  $u \leftarrow \mathcal{U}(\mathcal{S})$;
  for $i = 1$ *to* k **do**
    for $j = k+1$ *to* n **do**
      $\widehat{\mathcal{U}_{\pi^t(i),\pi^t(j)}} += u, m_{\pi^t(i),\pi^t(j)} += 1$;
for $i = 1$ *to* n **do**
  for $j = 1$ *to* n **do**
    $\widehat{\Delta\mathcal{SV}_{i,j}} \leftarrow \left(\frac{\widehat{\mathcal{U}_{i,j}}}{m_{i,j}} - \frac{\widehat{\mathcal{U}_{j,i}}}{m_{j,i}}\right)$;
return $\widehat{\Delta\mathcal{SV}_{1,2}}, \widehat{\Delta\mathcal{SV}_{1,3}}, \ldots, \widehat{\Delta\mathcal{SV}_{n-1,n}}$.

If we use Equation 6, for a sample coalition $\mathcal{S}$, the difference of utilities $\mathcal{U}(\mathcal{S}\cup\{z_i\}) - \mathcal{U}(\mathcal{S}\cup\{z_j\})$ can be used by only a pair of players, which is a situation similar to that of using marginal contributions in the Monte Carlo estimation methods using Equation 2. Using Equation 7, the utility $\mathcal{U}(\mathcal{S})$ can be used for all the differences $\Delta\mathcal{SV}_{i,j}$ with $z_i \in \mathcal{S}$ and $z_j \in \mathcal{N} \setminus \mathcal{S}$, and thus we can update $|\mathcal{S}|(n - |\mathcal{S}|)$ elements in the differential matrix using $\mathcal{U}(\mathcal{S})$ for one sample coalition $\mathcal{S}$.

Corollary 5.2 (Usage of utilities). *Using Algorithm 1, in expectation, each utility $\mathcal{U}(\mathcal{S})$ can be used to update $\frac{n^2}{(2\sum_{k=1}^{n-1}\frac{1}{k})}$ elements in the differential matrix.* □

In summary, the utility-based method is shown in Algorithm 1. To ensure unbiasedness, we set $p_k \propto \frac{1}{k(n-k)}$. We have the following result about the estimation quality.

Theorem 5.3 (Estimation using utilities). *Algorithm 1 provides unbiased estimators if every element in the differential matrix is updated at least once.* □

5.2 Stratified Monte Carlo Methods

Stratified sampling serves as an effective approach to improve the effectiveness of sampling in estimation, and often improves the efficiency of sampling-based approximation methods. With appropriate partitioning of the population and proper sample allocation, stratified sampling reduces the variance compared to unstratified sampling [54]. In this section, we explore stratified sampling strategies to estimate a differential matrix.

Naturally, we may stratify the coalitions based on the size, that is, we put the coalitions of the same size into the same stratum. In this way we have $n-1$ strata in total. We call a coalition with k players a k-coalition. Denote by $\mathfrak{S}^k$ the set of all k-coalitions. The utilities in the k^{th} stratum is $\{\mathcal{U}(\mathcal{S}) | \mathcal{S} \in \mathfrak{S}^k\}$. To explore the strategy of stratification, we define the following.

*Definition 5.4 (**Stratified utility**).* Given a set of players $\mathcal{N}$, for two different players z_i and z_j, denote by $\mathfrak{S}^k_{i\setminus j}$ the set of all k-coalitions containing z_i but not z_j. The **stratified utility** $\mathcal{U}^k_{i,j}$ for players z_i and z_j with coalition size k is the average of the utilities $\mathcal{U}(\mathcal{S})$ with $\mathcal{S} \in \mathfrak{S}^k_{i\setminus j}$, that is, $\mathcal{U}^k_{i,j} = \frac{1}{\binom{n-2}{k-1}} \sum_{\mathcal{S}\in\mathfrak{S}^k_{i\setminus j}} \mathcal{U}(\mathcal{S})$. Specifically, $\mathcal{U}^k_{i,i} = 0$. □

Based on Theorem 5.1 and Definition 5.4, we have the following result on the relationship between the differential matrix and stratified utility.

Corollary 5.5. *Given a set of players $\mathcal{N}$, the difference between the Shapley values of z_i and z_j is* $\Delta\mathcal{SV}_{i,j} = \frac{1}{n-1}\sum_{k=1}^{n-1}(\mathcal{U}^k_{i,j} - \mathcal{U}^k_{j,i})$. □

According to Definition 5.4, $\mathcal{U}^k_{i,j}$ is the expectation of utilities $\mathcal{U}(\mathcal{S})$ with $\mathcal{S} \in \mathfrak{S}^k_{i\setminus j}$. Consequently, estimating the differential matrix can be reformulated as a sampling process estimating stratified utilities according to Corollary 5.5.

To estimate $\mathcal{U}^k_{i,j}$ by sampling coalitions, given a set of coalition samples with size $m^k_{i,j}$ and the corresponding utilities, $\{\mathcal{U}(\mathcal{S}_1), \ldots, \mathcal{U}(\mathcal{S}_{m^k_{i,j}})\}$, where $\mathcal{S}_1, \ldots, \mathcal{S}_{m^k_{i,j}} \in \mathfrak{S}^k_{i\setminus j}$, the mean is $\widehat{\mathcal{U}^k_{i,j}} = \frac{1}{m^k_{i,j}} \sum_{l=1}^{m^k_{i,j}} \mathcal{U}(\mathcal{S}_l)$, which is an estimator of $\mathcal{U}^k_{i,j}$. Then, an estimator of $\Delta\mathcal{SV}_{i,j}$ is $\widehat{\Delta\mathcal{SV}_{i,j}} = \frac{1}{n-1}\sum_{k=1}^{n-1}\widehat{\mathcal{U}^k_{i,j}} - \frac{1}{n-1}\sum_{k=1}^{n-1}\widehat{\mathcal{U}^k_{j,i}}$.

Stratified sampling scales up the elements in the differential matrix we must estimate. To ensure unbiasedness, we design an approach that quickly populates the stratified difference matrix at the beginning of the algorithm. Specifically, for each stratum $\mathfrak{S}^k$ with $k \le \frac{n}{2}$, we select a random permutation of players $\{1, 2, \ldots, n\}$. We can generate $\lceil \frac{n}{k} \rceil$ disjoint coalitions covering the grand coalition. We consider the case $\mathcal{N} = \{1, 2, 3, 4, 5, 6\}$ and $k = 2$ for example. For a random permutation π of $\mathcal{N}$, we can derive 3 different coalitions $\{\pi(1), \pi(2)\}$, $\{\pi(3), \pi(4)\}$ and$\{\pi(5), \pi(6)\}$. For $k > \frac{n}{2}$, we generate $\lceil \frac{n}{n-k} \rceil$ complements of the coalitions that need to be sampled. Subsequently, we greedily fill in all the unsampled elements in the differential matrix (see details in Apendix A). In this way, we can guarantee that each $\mathcal{U}^k_{i,j}$ is sampled at least once. The method is given in Algorithm 2. To ensure variance reduction, we set p_k the same as Algorithm 1 [54].

Corollary 5.6 (Samples for the unbiased guarantee). *Algorithm 2 calls for $\Theta(n \log n)$ samples to make sure every element in the differential matrix is updated at least once.* □

Corollary 5.7 (Estimation using stratified utilities). *Algorithm 2 provides unbiased estimators.* □

5.3 Optimal Sample Allocation

One may be interested in the optimal sample allocation. Inspired by [7, 72], we develop the optimal sample allocation for Shapley value estimation based on the differential matrix. We have the following result.

Theorem 5.8 (Variances of the Estimators). *Consider the situation where each stratum $\mathfrak{S}^k$ is assigned with m_k samples in the Shapley value estimation, we have*

$$\mathrm{E}[\sum_{1\le i<j\le n} \mathrm{Var}(\widehat{\Delta\mathcal{SV}_{i,j}})] = \sum_{k=1}^{n-1} \frac{n(n-1)}{m_k k(n-k)} \sum_{i=1}^{n}\sum_{j=1}^{n} \sigma^2_{i,j,k},$$

where $\sigma^2_{i,j,k}$ is the variance of the set $\{\mathcal{U}(\mathcal{S})|\mathcal{S} \in \mathfrak{S}^k_{i\setminus j}\}$. □

Algorithm 2: Stratified differential matrix estimation using utilities.

input : players $\mathcal{N} = \{z_1, \ldots, z_n\}$ and $m \geq 4n \log n$

output: the estimated difference $\widehat{\Delta\mathcal{SV}_{i,j}}$

$\widehat{\mathcal{U}^k_{i,j}}, m^k_{i,j}, c \leftarrow 0$ $(1 \leq i, j \leq n, 1 \leq k \leq n-1)$;

// Ensure unbiasedness.

for $k = 1$ *to* $\lfloor \frac{n}{2} \rfloor$ **do**

  Let π^k be a random permutation of $\{1, \ldots, n\}$;

  for $l = 1$ *to* $\lceil \frac{n}{k} \rceil$ **do**

    $\mathcal{S} \leftarrow \{z_{\pi^k(k(l-1)+1)}, \ldots, z_{\pi^k(kl)}\}$;

    // If the index exceeds, perform a modulo n operation.

    $u \leftarrow \mathcal{U}(\mathcal{S})$;

    $nu \leftarrow \mathcal{U}(\mathcal{N} \setminus \mathcal{S})$;

    for $i = k(l-1) + 1$ *to* kl **do**

      for $j = kl + 1$ *to* $k(l-1) + n$ **do**

        $\widehat{\mathcal{U}^k_{\pi^t(i),\pi^t(j)}} += u, m^k_{\pi^t(i),\pi^t(j)} += 1$;

        $\widehat{\mathcal{U}^{n-k}_{\pi^t(j),\pi^t(i)}} += nu, m^{n-k}_{\pi^t(j),\pi^t(i)} += 1$;

        $c += 2$;

GreedyFill($\mathcal{N}$);

for $t = 1$ *to* $m - c$ **do**

  Select k with probability $p_k \propto \frac{1}{k(n-k)}$ $(1 \leq k \leq n-1)$;

  Let π^t be a random permutation of $\{1, \ldots, n\}$;

  $\mathcal{S} \leftarrow \{z_{\pi^t(1)}, \ldots, z_{\pi^t(k)}\}$;

  $u \leftarrow \mathcal{U}(\mathcal{S})$;

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

    for $j = k + 1$ *to* n **do**

      $\widehat{\mathcal{U}^k_{\pi^t(i),\pi^t(j)}} += u, m^k_{\pi^t(i),\pi^t(j)} += 1$;

for $i = 1$ *to* n **do**

  for $j = 1$ *to* n **do**

    $\widehat{\Delta\mathcal{SV}_{i,j}} \leftarrow \frac{1}{n-1} \sum_{k=1}^{n-1} \frac{\widehat{\mathcal{U}^k_{i,j}}}{m^k_{i,j}} - \frac{1}{n-1} \sum_{k=1}^{n-1} \frac{\widehat{\mathcal{U}^k_{j,i}}}{m^k_{j,i}}$;

return $\widehat{\Delta\mathcal{SV}_{1,2}}, \widehat{\Delta\mathcal{SV}_{1,3}}, \ldots, \widehat{\Delta\mathcal{SV}_{n-1,n}}$.

To minimize the variances of the estimators, we have the following result according to the equality condition of the Cauchy-Schwarz inequality.

Theorem 5.9. *For the objective function*

$$\min_{(m_1,\ldots,m_{n-1})^T \in \mathbb{N}^{n-1}} \mathrm{E}[\sum_{1 \leq i < j \leq n} \mathrm{Var}(\widehat{\Delta\mathcal{SV}_{i,j}})],$$

Algorithm 3: Stratified differential matrix estimation using utilities based on the optimal sample allocation.

input : players $\mathcal{N} = \{z_1, \ldots, z_n\}$ and $m, m_{\text{init}} \geq 4n \log n$
output: the estimated difference $\widehat{\Delta\mathcal{SV}_{i,j}}$

Call Algorithm 2 with $\mathcal{N}, m_{\text{init}}$;
Record each $\mathcal{U}(\mathcal{S})$ to corresponding $\mathfrak{S}^k_{i\setminus j}$;
// Bessel's correction.
for $i = 1$ *to* n **do**
  for $j = 1$ *to* n **do**
    for $k = 1$ *to* $n-1$ **do**
      $u_m \leftarrow \frac{1}{m_{i,j,k}} \sum_{u \in \mathfrak{S}^k_{i\setminus j}} u$;
      $\hat{\sigma}^2_{i,j,k} = \frac{1}{m_{i,j,k}-1} \sum_{u \in \mathfrak{S}^k_{i\setminus j}} (u - u_m)^2$;
for $k = 1$ *to* $n-1$ **do**
  $w_k \leftarrow \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} \frac{\hat{\sigma}^2_{i,j,k}}{k(n-k)}}$;
for $t = 1$ *to* $m - m_{init}$ **do**
  Select k with probability $p_k \propto w_k$ $(1 \leq k \leq n-1)$;
  Let π^t be a random permutation of $\{1, \ldots, n\}$;
  $\mathcal{S} \leftarrow \{z_{\pi^t(1)}, \ldots, z_{\pi^t(k)}\}$;
  $u \leftarrow \mathcal{U}(\mathcal{S})$;
  for $i = 1$ *to* k **do**
    for $j = k+1$ *to* n **do**
      $\widehat{\mathcal{U}^k_{\pi^t(i),\pi^t(j)}} \mathrel{+}= u, m^k_{\pi^t(i),\pi^t(j)} \mathrel{+}= 1$;
for $i = 1$ *to* n **do**
  for $j = 1$ *to* n **do**
    $\widehat{\Delta\mathcal{SV}_{i,j}} \leftarrow \frac{1}{n-1} \sum_{k=1}^{n-1} \frac{\widehat{\mathcal{U}^k_{i,j}}}{m^k_{i,j}} - \frac{1}{n-1} \sum_{k=1}^{n-1} \frac{\widehat{\mathcal{U}^k_{j,i}}}{m^k_{j,i}}$;
return $\widehat{\Delta\mathcal{SV}_{1,2}}, \widehat{\Delta\mathcal{SV}_{1,3}}, \ldots, \widehat{\Delta\mathcal{SV}_{n-1,n}}$.

$$s.t. \quad \sum_{i=1}^{n-1} m_k = m.$$

where m_k is the sample size assigned to $\mathfrak{S}^k$. We have

$$m_k \propto \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} \frac{\sigma^2_{i,j,k}}{k(n-k)}}.$$

However, it is challenging to compute the variance of the utilities in each stratum. To this end, we first invoke Algorithm 2 to estimate the variance of the utilities in each stratum based on Bessel's correction. Subsequently, we use the variances to compute a proper sample allocation. The method is given in Algorithm 3.

Algorithm 4: Partial differential matrix estimation using utilities

input : a set of players $\mathcal{N} = \{z_1, \ldots, z_n\}$ and sample size $m > 0$
output: the estimated difference $\widehat{\Delta\mathcal{SV}_{1,i}}$

$\widehat{\mathcal{U}_{1,i}}, \widehat{\mathcal{U}_{i,1}}, m_{1,i}, m_{i,1} \leftarrow 0$ $(2 \le i \le n)$;
for $t = 1$ *to* m **do**
  Select k with probability $p_k \propto \frac{1}{k(n-k)}$ $(1 \le k \le n-1)$;
  Let π^t be a random permutation of $\{1, \ldots, n\}$;
  $\mathcal{S} \leftarrow \{z_{\pi^t(1)}, \ldots, z_{\pi^t(k)}\}$;
  $u \leftarrow \mathcal{U}(\mathcal{S})$;
  if $1 \in \mathcal{S}$ **then**
    for $i = k+1$ *to* n **do**
      $\widehat{\mathcal{U}_{1,\pi^t(i)}} += u, m_{1,\pi^t(i)} += 1$;
  else
    for $i = 1$ *to* k **do**
      $\widehat{\mathcal{U}_{\pi^t(i),1}} += u, m_{\pi^t(i),1} += 1$;
for $i = 2$ *to* n **do**
  $\widehat{\Delta\mathcal{SV}_{1,i}} \leftarrow \left(\frac{\widehat{\mathcal{U}_{1,i}}}{m_{1,i}} - \frac{\widehat{\mathcal{U}_{i,1}}}{m_{i,1}}\right)$;
return $\widehat{\Delta\mathcal{SV}_{1,2}}, \widehat{\Delta\mathcal{SV}_{1,3}}, \ldots, \widehat{\Delta\mathcal{SV}_{1,n}}$.

6 Theoretical Analysis

In a fully estimated differential matrix, every element is estimated using some samples. In a partially estimated differential matrix, some elements may not be estimated by any samples and thus just take the value of 0 for convenience. In this section, we provide a theoretical analysis to show the superiority of estimating the Shapley values with a fully estimated differential matrix compared to using a partially estimated differential matrix (Theorem 6.1) and the state-of-the-art methods based on complementary contributions and utilities (Theorems 6.3 and 6.4).

The Shapley value calculation only requires a partially estimated differential matrix. For instance, we only need the first row which represents the differences between the Shapley values of z_1 and the other players due to the triangularity $\Delta\mathcal{SV}_{i,j} = \Delta\mathcal{SV}_{1,j} - \Delta\mathcal{SV}_{1,i}$. This approach remains intuitive even with an estimated differential matrix. The corresponding algorithms for partial differential matrix estimation are presented in Algorithms 4 and 5 (unstratified and stratified). However, the estimated differential matrix does not exhibit the triangularity because utilities do not hold the triangularity. In such cases, using a fully estimated differential matrix for Shapley value estimation is advantageous owing to its more comprehensive information. Moreover, the set of utilities for estimating the first row is identical to the fully estimated differential matrix. Thus, the fully estimated differential matrix does not require extra samples compared to the partially estimated differential matrix.

Let us consider two approaches. Firstly, in **Approach *P***, using the partially estimated differential matrix (e.g., the first row of the differential matrix) and the efficiency axiom to estimate Shapley

Algorithm 5: Stratified partial differential matrix estimation using utilities

input : a set of players $\mathcal{N} = \{z_1, \ldots, z_n\}$ and sample size $m \geq 4n \log n$

output : the estimated difference $\widehat{\Delta\mathcal{SV}_{1,i}}$

$\widehat{\mathcal{U}^k_{1,i}}, \widehat{\mathcal{U}^k_{i,1}}, m^k_{1,i}, m^k_{i,1} \leftarrow 0$ $(2 \leq i \leq n, 1 \leq k \leq n-1)$;

// Ensure unbiasedness.

$\widehat{\mathcal{U}^1_{1,i}} = \mathcal{U}(\{z_1\}), m^1_{1,i} = 1$ $(2 \leq i \leq n)$;

$\widehat{\mathcal{U}^{n-1}_{i,1}} = \mathcal{U}(\mathcal{N} \setminus \{z_1\}), m^{n-1}_{i,1} = 1, c \leftarrow 2$ $(2 \leq i \leq n)$;

for $k = 1$ *to* $\lfloor \frac{n-1}{2} \rfloor$ **do**

  Let π^k be a random permutation of $\{2, \ldots, n\}$;

  for $l = 1$ *to* $\lceil \frac{n-1}{k} \rceil$ **do**

    $\mathcal{S} \leftarrow \{z_{\pi^k(k(l-1)+1)}, \ldots, z_{\pi^k(kl)}\}$;

    $u \leftarrow \mathcal{U}(\mathcal{S}), u_1 \leftarrow \mathcal{U}(\mathcal{S} \cup \{z_1\})$;

    $nu \leftarrow \mathcal{U}(\mathcal{N} \setminus \mathcal{S}), nu_1 \leftarrow \mathcal{U}(\mathcal{N} \setminus (\mathcal{S} \cup \{z_1\}))$;

    for $i = k(l-1) + 1$ *to* kl **do**

      $\widehat{\mathcal{U}^k_{\pi^t(i),1}} \mathrel{+}= u, m^k_{\pi^t(i),1} \mathrel{+}= 1$;

      $\widehat{\mathcal{U}^{n-k}_{1,\pi^t(i)}} \mathrel{+}= nu, m^{n-k}_{1,\pi^t(i)} \mathrel{+}= 1$;

    for $i = kl + 1$ *to* $k(l-1)$ **do**

      $\widehat{\mathcal{U}^{k+1}_{1,\pi^t(i)}} \mathrel{+}= u_1, m^{k-1}_{1,\pi^t(i)} \mathrel{+}= 1$;

      $\widehat{\mathcal{U}^{n-k-1}_{\pi^t(i),1}} \mathrel{+}= nu_1, m^{n-k-1}_{\pi^t(i),1} \mathrel{+}= 1$;

    $c \mathrel{+}= 4$;

for $t = 1$ *to* $m - c$ **do**

  Select k with probability $p_k \propto \frac{1}{k(n-k)}$ $(2 \leq k \leq n-2)$;

  Let π^t be a random permutation of $\{1, \ldots, n\}$;

  $\mathcal{S} \leftarrow \{z_{\pi^t(1)}, \ldots, z_{\pi^t(k)}\}$;

  $u \leftarrow \mathcal{U}(\mathcal{S})$;

  if $1 \in \mathcal{S}$ **then**

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

      $\widehat{\mathcal{U}^k_{1,\pi^t(i)}} \mathrel{+}= u, m^k_{1,\pi^t(i)} \mathrel{+}= 1$;

  else

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

      $\widehat{\mathcal{U}^k_{\pi^t(i),1}} \mathrel{+}= u, m^k_{\pi^t(i),1} \mathrel{+}= 1$;

for $i = 2$ *to* n **do**

  $\widehat{\Delta\mathcal{SV}_{1,i}} \leftarrow \frac{1}{n-1} \sum_{k=1}^{n-1} \frac{\widehat{\mathcal{U}^k_{1,i}}}{m^k_{1,i}} - \frac{1}{n-1} \sum_{k=1}^{n-1} \frac{\widehat{\mathcal{U}^k_{i,1}}}{m^k_{i,1}}$;

return $\widehat{\Delta\mathcal{SV}_{1,2}}, \widehat{\Delta\mathcal{SV}_{1,3}}, \ldots, \widehat{\Delta\mathcal{SV}_{1,n}}$.

values, the estimated Shapley value $\widehat{\mathcal{SV}^p_i}$ is

$$\widehat{\mathcal{SV}^p_1} = \frac{1}{n} \sum_{j=1}^{n} \widehat{\Delta\mathcal{SV}_{1,j}} + \frac{1}{n}(\mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset)),$$

$$\widehat{\mathcal{SV}_i^P} = \widehat{\mathcal{SV}_1^P} - \widehat{\Delta\mathcal{SV}_{1,i}} \quad (i \neq 1).$$

In **Approach *F***, using the fully estimated differential matrix and the efficiency axiom to estimate Shapley values, the estimated Shapley value $\widehat{\mathcal{SV}_i^F}$ is

$$\widehat{\mathcal{SV}_i^F} = \frac{1}{n}\sum_{j=1}^{n}\widehat{\Delta\mathcal{SV}_{i,j}} + \frac{1}{n}(\mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset)).$$

Approach *F* reconciles with Theorem 4.3 in Section 4.3. We show that Approach *F* produces an expected sum of variances on estimated Shapley values equal to or smaller than Approach *P* when $n \geq 5$.

Theorem 6.1.
When $n \geq 5$, $\mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i^F})] \leq \mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i^P})]$. □

The expected sum of variances of the estimated Shapley values based on the differential matrix is given by the following.

Theorem 6.2. *With m samples, the expected sum of variances of the estimated Shapley values based on the differential matrix using Algorithm 1 is*

$$\mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i^F})] = \frac{2(n-1)\sum_{k=1}^{n-1}\frac{1}{k}}{nm}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\sigma_{i,j}^2 + \sum_{l=1}^{n}\sigma_{i,j,l}^2\right),$$

where $\sigma_{i,j}$ *is the variance of the random variable over the set* $\{\mathcal{U}(\mathcal{S})|z_i \in \mathcal{S}, z_j \notin \mathcal{S}\}$ *with* $Pr(\mathcal{U} = \mathcal{U}(\mathcal{S})) = \frac{1}{\binom{n-2}{|\mathcal{S}|-1}}$, *and* $\sigma_{i,j,l}$ *is the covariance of the of the random variables over the set* $\{\mathcal{U}(\mathcal{S})|z_i \in \mathcal{S}, z_j \notin \mathcal{S}\}$ *and the set* $\{\mathcal{U}(\mathcal{S})|z_i \in \mathcal{S}, z_l \notin \mathcal{S}\}$, *respectively, with* $Pr(\mathcal{U} = \mathcal{U}(\mathcal{S})) = \frac{1}{\binom{n-2}{|\mathcal{S}|-1}}$.

For Algorithm 2,

$$\mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i^F})] = \frac{2(n-1)\sum_{k=1}^{n-1}\frac{1}{k}}{nm}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n-1}\left(\sigma_{i,j,k}^2 + \sum_{l=1}^{n}\sigma_{i,j,l,k}^2\right),$$

where $\sigma_{i,j,l,k}$ *is the covariance of the uniformly distributed random variables over the set* $\{\mathcal{U}(\mathcal{S})|\mathcal{S} \in \mathfrak{S}_{i\backslash j}^k\}$ *and the set* $\{\mathcal{U}(\mathcal{S})|\mathcal{S} \in \mathfrak{S}_{i\backslash l}^k\}$, *respectively.* □

Based on Theorem 6.2, we can conduct a comparative analysis between Approach *F* and the state-of-the-art methods based on complementary contributions or utilities, demonstrating Approach *F* can guarantee an equal or smaller expected sum of variances.

Theorem 6.3. *Given a set of players* $\mathcal{N}$, *let* $\widehat{\mathcal{SV}_i^U}$ *be the estimated Shapley value for player* z_i *as computed by Li and Yu [37]. Denote by* $\rho_m^U = \mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i^F})]/\mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i^U})]$ *with m samples. We have* $\lim_{m\to\infty}\rho_m^U = c_{\mathcal{U}}$, *where we use Algorithm 1 for the differential matrix estimation and* $c_{\mathcal{U}} \leq 1 - 1/n$ *is a constant based on the utility function. Specifically, with the condition* $2\mathrm{Cov}(\widehat{\mathcal{SV}_i^U}, \widehat{\mathcal{SV}_j^U}) > c_0(\mathrm{Var}(\widehat{\mathcal{SV}_i^U}) + \mathrm{Var}(\widehat{\mathcal{SV}_j^U}))$ *for any arbitrage players* z_i *and* z_j, *we have* $c_{\mathcal{U}} \leq (1-1/n)(1-c_0)$. □

Theorem 6.4. *Given a set of players* $\mathcal{N}$, *let* $\widehat{\mathcal{SV}_i^C}$ *be the estimated Shapley value for player* z_i *as computed by Zhang et al. [72]. Denote by* $\rho_m^C = \mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i^F})]/\mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i^C})]$ *with m samples. We have* $\lim_{m\to\infty}\rho_m^C = c_C$, *where we use Algorithm 1 for the differential matrix estimation and* $c_C \leq 1 - 1/n$ *is a constant based on the utility function with the condition that if* $\mathcal{U}(\mathcal{S}_1) \geq \mathcal{U}(\mathcal{S}_2)$, *we*

have $\mathcal{U}(\mathcal{N}\setminus\mathcal{S}_1) \leq \mathcal{U}(\mathcal{N}\setminus\mathcal{S}_2)$. *Specifically, with the condition* $2\mathrm{Cov}(\widehat{\mathcal{SV}_i^C}, \widehat{\mathcal{SV}_j^C}) > c_0(\mathrm{Var}(\widehat{\mathcal{SV}_i^C}) + \mathrm{Var}(\widehat{\mathcal{SV}_j^C}))$ *for any arbitrage players* z_i *and* z_j, *we have* $c_{\mathcal{U}} \leq (1-1/n)(1-c_0)$. □

In general, the covariance between Shapley value estimators of [37, 72] is appreciable as [37, 72] reuses each sample and each Shapley value estimator receives a substantial portion of duplicate samples. Therefore, c_0 is a non-trivial value.

Interpretation of superiority. Approach F yields an equal or smaller expected sum of variances, generally leading to equal or lower estimation error (e.g., root mean square error). Here we show how to connect the expected sum of variances with the estimation error for Shapley values. The expected sum of variances for estimated Shapley values is defined as

$$\mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i})] = \mathrm{E}[\sum_{i=1}^{n}\mathrm{E}(\widehat{\mathcal{SV}_i} - \mathrm{E}[\widehat{\mathcal{SV}_i}])^2].$$

For the unbiased estimator $\widehat{\mathcal{SV}_i}$, that is, $\mathrm{E}[\widehat{\mathcal{SV}_i}] = \mathcal{SV}_i$, we have

$$\mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i})] = \mathrm{E}[\sum_{i=1}^{n}\mathrm{E}(\widehat{\mathcal{SV}_i} - \mathcal{SV}_i)^2],$$

which measures the expected sum of square errors. Therefore,
$\mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i})]$ can be used as an appropriate evaluation metric for unbiased Shapley value estimation.

Complexity. We analyze the complexity of the proposed algorithms.

Theorem 6.5. *Algorithms 1 - 5 require* $O(\frac{n}{\varepsilon^2}\log\frac{n}{\delta})$ *samples to reach an* (ε,δ)*-approximation in RMSE (see Section 7).* □

Space and time cost. We analyze the space and time cost of the proposed algorithms.

Theorem 6.6. *The space consumption of Algorithm 4 is* $O(n)$*; the space consumption of Algorithms 1 and 5 is* $O(n^2)$*; the space consumption of Algorithms 2 and 3 is* $O(n^3)$. *The runtime per sample for Algorithms 4 and 5 is* $O(n+u(n))$*; the runtime per sample for Algorithms 1 and 2 is* $O(\frac{n^2}{\log n}+u(n))$*; the runtime per sample for Algorithms 3 is* $O(n^2+u(n))$, *where* $u(n)$ *is the time complexity of the utility function* $\mathcal{U}(\cdot)$ *with n players.* □

7 Experiments

In this section, we present experimental results evaluating the effectiveness and efficiency of the proposed algorithms for Shapley value estimation.

7.1 Experiment Setup

We conduct experiments on a server with two Intel(R) Xeon(R) Platinum 8383C CPUs @ 2.70GHz and 256GB memory, running Ubuntu 20.04.6 LTS.

7.1.1 Methods Compared. We compare our proposed algorithms with three representative algorithms based on sampling [8, 37, 72] and two representative algorithms based on optimization [13, 44].
The sampling-based methods include

- **MC** [8]: the Monte Carlo simulation algorithm based on marginal contributions.
- **GELS** [37]: the Monte Carlo simulation algorithm based on utilities.
- **CC** [72]: the Monte Carlo simulation algorithm based on complementary contributions.

The optimization-based methods include

- **KernelSHAP** [44]: the kernel-based optimization method based on utility sampling.
- **PairedSHAP** [13]: the enhanced algorithm of KernelSHAP based on pairwise utility sampling.

We compare five versions of the methods proposed in this paper.

- **Diff**: the proposed algorithm based on the differential matrix with the unstratified strategy using Approach F (Algorithm 1).
- **S-Diff**: the proposed algorithm based on the differential matrix with the stratified strategy using Approach F (Algorithm 2).
- **S-Diff+**: the proposed algorithm based on the differential matrix with the suboptimal stratified strategy (Algorithm 3).
- **Diff-**: the proposed algorithm based on the differential matrix with the unstratified strategy using Approach P (Algorithm 4).
- **S-Diff-**: the proposed algorithm based on the differential matrix with the stratified strategy using Approach P (Algorithm 5).

7.1.2 Evaluation Tasks. To demonstrate the effectiveness and efficiency of the proposed algorithms, we estimate Shapley values and conduct empirical analysis in four distinct application scenarios including two cooperative games and two data valuation tasks.

***A voting game** [55].* In a non-symmetric voting game simulating a U.S. presidential election designed by Owen [55], players vote with the principle of majority rule. The Shapley value can be regarded as a metric for evaluating the voting influence of each player. The set of players for a voting game is $\mathcal{N} = \{z_1, \ldots, z_{51}\}$. For any coalition $\mathcal{S}$ ($\mathcal{S} \subseteq \mathcal{N}$), the utility function for a voting game is

$$\mathcal{U}_v(\mathcal{S}) = \begin{cases} 1, & \text{if } \sum_{z_i \in \mathcal{S}} w_i \geq \frac{1}{2} \sum_{z_j \in \mathcal{N}} w_j, \\ 0, & \text{otherwise,} \end{cases}$$

where $\{w_1, \ldots, w_{51}\} = \{45, 41, 27, 26, 26, 25, 21, 17, 17, 14, 13, 13, 12, 12, 12, 11, \underbrace{10, \ldots, 10}_{4}, \underbrace{9, \ldots, 9}_{4}, 8, 8, \underbrace{7, \ldots, 7}_{4}, \underbrace{6, \ldots, 6}_{4}, 5, \underbrace{4, \ldots, 4}_{9}, \underbrace{3, \ldots, 3}_{7}\}$ denote the weights of votes.

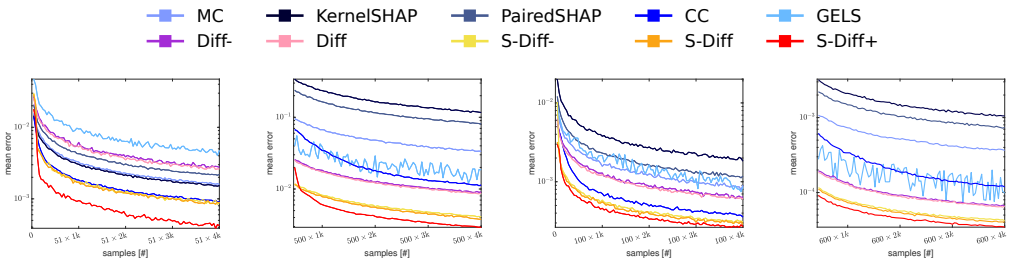
***An airport game** [40].* In an airport game for airstrip cost allocation, players (i.e., planes) need to share the cost determined by the maximal size of the planes. The Shapley value is the established solution to distribute the airstrip cost among different planes of various sizes equitably. The set of players for an airport game is $\mathcal{N} = \{z_1, \ldots, z_{500}\}$. For any coalition $\mathcal{S}$ ($\mathcal{S} \subseteq \mathcal{N}$), the utility function for an airport game is

$$v_a(\mathcal{S}) = \max_{z_i \in \mathcal{S}} \{c_i\},$$

where c_i is the cost of player z_i when building airstrip exclusively and $c_1, \ldots, c_{500}$ are randomly generated integers in $[1, 100]$.

We subsequently introduce two data valuation tasks. In a data valuation task, players are the data points used in the training of a machine learning model. The Shapley value is utilized to evaluate the contribution of each player to the model's utility.

***The Iris data valuation task**.* Consider a ternary classification data valuation task for Iris dataset prediction, where we employ a support vector machine (SVM) classifier as the predictive model and the prediction accuracy on a validation set as the utility function. Specifically, we employ the Iris dataset from the UCI machine learning repository [21] in this task. A subset of 100 data points is randomly selected for training as the set of players $\mathcal{N}$, and the remaining 50 points are



(a) Voting. (b) Airport. (c) Iris. (d) Breast Cancer.

Fig. 1. Shapley value computation effectiveness (mean error).

used for validation. The utility function for a coalition $\mathcal{S}$ is defined as the validation accuracy of the model trained over $\mathcal{S}$.

The Breast Cancer data valuation task. Consider a binary classification data valuation task for breast cancer prediction, where we employ a support vector machine classifier as the predictive model and the prediction accuracy on a validation set as the utility function. Specifically, we employ the Breast Cancer Wisconsin dataset from the UCI machine learning repository [68] in this task. A subset of 600 data points is randomly selected for training as the set of players $\mathcal{N}$, and the remaining 99 points are used for validation. The utility function for a coalition $\mathcal{S}$ is defined as the validation accuracy of the model trained over $\mathcal{S}$.

7.1.3 *Evaluation Metrics.*

Mean error. Given the benchmark Shapley values $\mathcal{SV}_i$ and the estimated Shapley values $\widehat{\mathcal{SV}_i}$ $(i = 1, \ldots, n)$, the mean error for the estimated Shapley values compared to the benchmark Shapley values is

$$\text{mean error} = \frac{1}{n}\sum_{i=1}^{n}\left|\widehat{\mathcal{SV}_i} - \mathcal{SV}_i\right|.$$

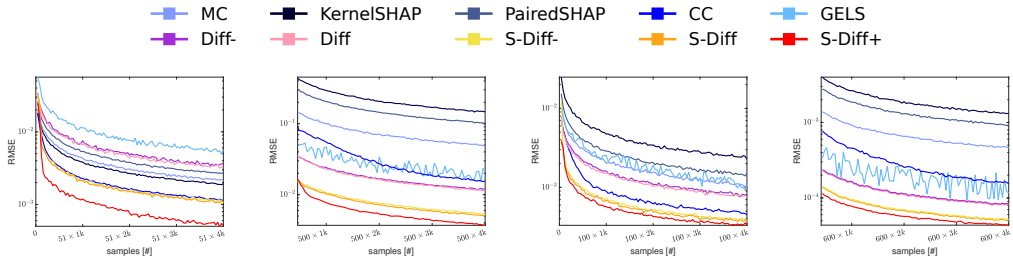
Root mean square error (RMSE). Given the benchmark Shapley values $\mathcal{SV}_i$ and the estimated Shapley values $\widehat{\mathcal{SV}_i}$ $(i = 1, \ldots, n)$, the root mean square error for the estimated Shapley values compared to the benchmark Shapley values is

$$\text{RMSE} = \left(\frac{1}{n}\sum_{i=1}^{n}\left|\widehat{\mathcal{SV}_i} - \mathcal{SV}_i\right|^2\right)^{1/2}.$$

Maximum error. Given the benchmark Shapley values $\mathcal{SV}_i$ and the estimated Shapley values $\widehat{\mathcal{SV}_i}$ $(i = 1, \ldots, n)$, the maximum error for the estimated Shapley values compared to the benchmark Shapley values is

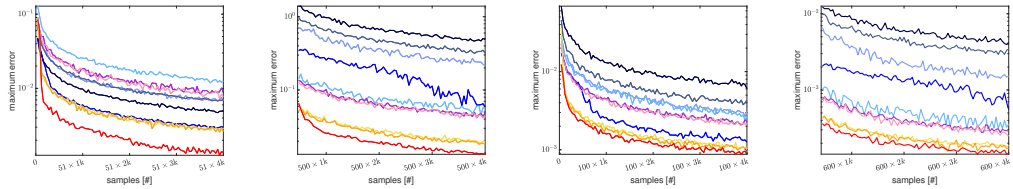
$$\text{maximum error} = \max_i\left|\widehat{\mathcal{SV}_i} - \mathcal{SV}_i\right|.$$

Computing the exact Shapley value $\mathcal{SV}_i$ for evaluation purposes is prohibitively expensive because the computation cost grows exponentially with the number of players. Therefore, we use the estimated Shapley value computed by the classic Monte Carlo simulation algorithm based on complementary contributions [72] with 500k samples as the benchmark Shapley value for the data valuation tasks in Figures 1-3. For the cooperative games, we employ the Shapley value reported by Castro et al. [8] for the voting game and those computed by Littlechild and Owen [39] for the airport game as the benchmark Shapley values.



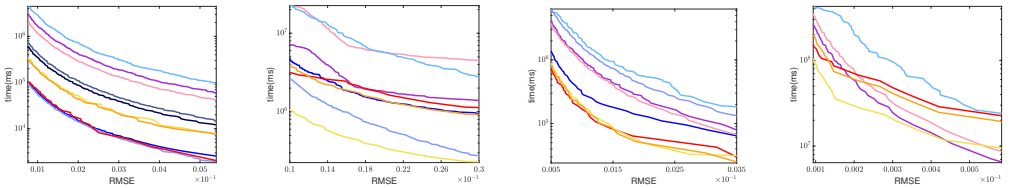
(a) Voting. (b) Airport. (c) Iris. (d) Breast Cancer.

Fig. 2. Shapley value computation effectiveness (RMSE).



(a) Voting. (b) Airport. (c) Iris. (d) Breast Cancer.

Fig. 3. Shapley value computation effectiveness (maximum error).

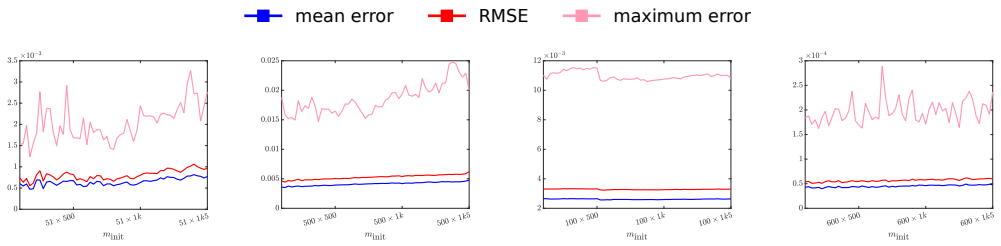


(a) Voting. (b) Airport. (c) Iris. (d) Breast Cancer.

Fig. 4. Shapley value computation efficiency.

7.2 Effectiveness

We experimentally analyze the performance of Diff-, Diff, S-Diff-, S-Diff, and S-Diff+ with the same number of samples across four tasks: Voting, Airport, Iris, and Breast Cancer. Specifically, we set $m_{\text{init}} = \frac{m}{2}$ for S-Diff+ in Sections 7.2 and 7.3. The empirical results, shown in Figures 1-3 consistently reveal the superiority of the proposed methods based on the differential matrix compared to the baselines across all essential metrics including mean error, RMSE, and maximum error. Particularly, RMSE, S-Diff-, S-Diff, and S-Diff+ show stable superiority compared to the state-of-the-art baselines CC and GELS, reconciling with Theorems 6.3 and 6.4 since RMSE is an estimator for the standard deviation of the estimated Shapley values. Moreover, Diff and S-Diff consistently outperform Diff- and S-Diff, reconciling with Theorem 6.1. Our methods are also highly robust as the errors steadily decline as the number of samples increases. Diff- and Diff performs relatively better in large-scale applications since the number of strata is larger in such a case and each stratum receives relatively fewer samples. S-Diff-, S-Diff, and S-Diff+ suffer from insufficient samples in such a case. Notably, due to the instability of the maximum error, the trend of smaller errors is significantly perturbed in Figure 3 as the number of samples increases.



(a) Voting. (b) Airport. (c) Iris. (d) Breast Cancer.

Fig. 5. Effect of m_{init}.

7.3 Efficiency

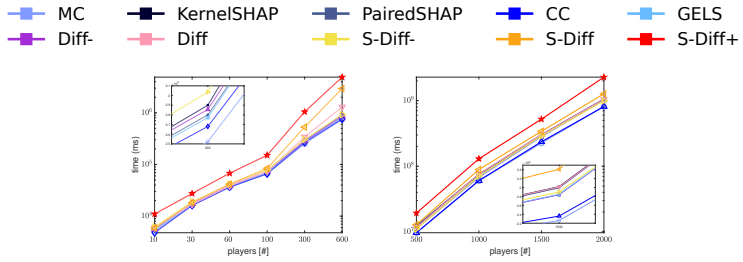
According to Section 7.2, RMSE is a relatively stable metric for estimation quality. Therefore, it is natural to evaluate the efficiency of algorithms by exploring the required time to achieve a target RMSE. Specifically, we conduct an empirical analysis across four various tasks with various target RMSE in this section. Figure 4 shows the required time of each algorithm to achieve the same target RMSE. We omit some underperforming baselines in each figure for easy reading. In cooperative games, existing methods with much simpler logic MC and CC exhibit higher efficiency, while Diff, S-Diff, and S-Diff+ offer limited improvement. As stated in Theorem 6.6, each sample in Diff, S-Diff, and S-Diff+ is used to update $O(\frac{n^2}{\log n})$ elements, while the time cost for computing utility functions in cooperative games is relatively small. The higher time cost for updating elements rather than computing utility functions in cooperative games leads to less efficiency of Diff, S-Diff, and S-Diff+. However, in the data valuation task, where model training is the primary time cost per sample, S-Diff and S-Diff+ require less time than baseline methods to reach the same RMSE level. As the number of players increases, we observe that Diff- and S-Diff- outperform Diff, S-Diff, and S-Diff+ in the airport and breast cancer tasks, as they only need to update $O(n)$ elements per sample on average, making them more efficient.

7.4 Effect of m_{init}

In Algorithm 3, we use samples of size m_{init} to estimate the variance of the utilities in each stratum and then compute a sample allocation. Here, we analyze the effect of m_{init}. We run an experiment of $2000 \times n$ samples for each game and set m_{init} from $100 \times n$ to $1500 \times n$ for S-Diff+. Figure 5 shows the mean error, RMSE, and the maximum error of S-Diff+ for each game with different m_{init}. Overall, the impact of m_{init} on the results is relatively minor, but it can be observed that the error gradually increases as m_{init} grows. One possible explanation is that S-Diff+ requires only a small number of samples to compute an effective sample allocation. As m_{init} increases, fewer samples are applied to this suboptimal sample allocation, gradually increasing error.

7.5 Scalability

To evaluate our proposed algorithms in large-scale scenarios, we present the runtime of each algorithm with a fixed sample size rather than error or RMSE, as obtaining an accurate benchmark for large scales within a meaningful time is not feasible. For the breast cancer task, we vary the number of players to 10, 30, 60, 100, 300, and 600. For larger scales, we use a real Abalone dataset from the UCI machine learning repository [53], randomly sampling 500, 1000, 1500, and 2000 data points for training and computing the Shapley values based on a test set of 200 data points. Figure 6 presents the runtime of each algorithm with $1000 \times n$ samples as the number of players increases. The time cost of our proposed algorithms is generally higher than that of MC and CC with the



(a) Breast Cancer. (b) Abalone.

Fig. 6. Shapley value computation scalability.

sample size. Nevertheless, our algorithms should achieve a better approximation, showing lower error and RMSE with the same runtime. Owing to fewer updates of elements, Diff- and S-Diff- exhibit similar time costs to GELS, which performs better than CC in large-scale scenarios. Similarly, Diff shows time costs comparable to GELS owing to lower space costs and faster updates. As the scale increases beyond 2000, S-Diff and S-Diff+ may encounter memory issues due to the space cost of $O(n^3)$, while Diff-, Diff, and S-Diff- remain efficient.

8 Conclusion

Our study introduces a novel method for Shapley value computation that leverages the efficiency axiom to achieve lower variances than existing methods. By utilizing the innovative concept of the differential matrix and employing a least-squares optimization approach, we provide more accurate estimates. Our Monte Carlo methods, including a stratified approach, further reduce variances. The superiority of the proposed method using the efficiency axiom is confirmed through mathematical analysis and experimental validation, opening a new direction for Shapley value computation in large-scale applications.

Acknowledgments

J. Pang, X. Li, and J. Liu are supported in part by the National Key RD Program of China (No. 2022YFB3103401), NSFC (No. 62102352, 62472378, and U23A20306), and the Zhejiang Province Pioneer Plan (No. 2024C01074).

References

[1] Anish Agarwal, Munther A. Dahleh, and Tuhin Sarkar. 2019. A marketplace for data: An algorithmic solution. In *Proceedings of the 2019 ACM Conference on Economics and Computation, EC 2019, Phoenix, AZ, USA, June 24-28, 2019*, Anna Karlin, Nicole Immorlica, and Ramesh Johari (Eds.). ACM, 701–726. https://doi.org/10.1145/3328526.3329589

[2] Dana Arad, Daniel Deutch, and Nave Frost. 2024. Predicting Fact Contributions from Query Logs with Machine Learning. In *Proceedings 27th International Conference on Extending Database Technology, EDBT 2024, Paestum, Italy, March 25 - March 28*, Letizia Tanca, Qiong Luo, Giuseppe Polese, Loredana Caruccio, Xavier Oriol, and Donatella Firmani (Eds.). OpenProceedings.org, 704–716. https://doi.org/10.48786/EDBT.2024.60

[3] Santiago Andrés Azcoitia, Costas Iordanou, and Nikolaos Laoutaris. 2023. Understanding the Price of Data in Commercial Data Marketplaces. In *39th IEEE International Conference on Data Engineering, ICDE 2023, Anaheim, CA, USA, April 3-7, 2023*. IEEE, 3718–3728. https://doi.org/10.1109/ICDE55515.2023.00300

[4] Leopoldo E. Bertossi, Benny Kimelfeld, Ester Livshits, and Mikaël Monet. 2023. The Shapley Value in Database Management. *SIGMOD Rec.* 52, 2 (2023), 6–17. https://doi.org/10.1145/3615952.3615954

[5] Ranran Bian, Yun Sing Koh, Gillian Dobbie, and Anna Divoli. 2019. Identifying Top-k Nodes in Social Networks: A Survey. *ACM Comput. Surv.* 52, 1 (2019), 22:1–22:33. https://doi.org/10.1145/3301286

[6] Meghyn Bienvenu, Diego Figueira, and Pierre Lafourcade. 2024. When is Shapley Value Computation a Matter of Counting? *PODS* 2, 2 (2024), 105. https://doi.org/10.1145/3651606

[7] Javier Castro, Daniel Gómez, Elisenda Molina, and Juan Tejada. 2017. Improving polynomial estimation of the Shapley value by stratified random sampling with optimum allocation. *Comput. Oper. Res.* 82 (2017), 180–188. https://doi.org/10.1016/J.COR.2017.01.019

[8] Javier Castro, Daniel Gómez, and Juan Tejada. 2009. Polynomial calculation of the Shapley value based on sampling. *Comput. Oper. Res.* 36, 5 (2009), 1726–1730. https://doi.org/10.1016/J.COR.2008.04.004

[9] Lingjiao Chen, Paraschos Koutris, and Arun Kumar. 2019. Towards Model-based Pricing for Machine Learning in a Data Marketplace. In *Proceedings of the 2019 International Conference on Management of Data, SIGMOD'19, Amsterdam, The Netherlands, June 30 - July 5, 2019*, Peter A. Boncz, Stefan Manegold, Anastasia Ailamaki, Amol Deshpande, and Tim Kraska (Eds.). ACM, 1535–1552. https://doi.org/10.1145/3299869.3300078

[10] Lingjiao Chen, Hongyi Wang, Leshang Chen, Paraschos Koutris, and Arun Kumar. 2019. Demonstration of Nimbus: Model-based Pricing for Machine Learning in a Data Marketplace. In *Proceedings of the 2019 International Conference on Management of Data, SIGMOD'19, Amsterdam, The Netherlands, June 30 - July 5, 2019*, Peter A. Boncz, Stefan Manegold, Anastasia Ailamaki, Amol Deshpande, and Tim Kraska (Eds.). ACM, 1885–1888. https://doi.org/10.1145/3299869.3320231

[11] Yiwei Chen, Kaiyu Li, Guoliang Li, and Yong Wang. 2024. Contributions Estimation in Federated Learning: A Comprehensive Experimental Evaluation. *Proceedings of the VLDB Endowment* 17, 8 (2024), 2077–2090.

[12] Shay B. Cohen, Eytan Ruppin, and Gideon Dror. 2005. Feature Selection Based on the Shapley Value. In *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30 - August 5, 2005*, Leslie Pack Kaelbling and Alessandro Saffiotti (Eds.). Professional Book Center, 665–670. http://ijcai.org/Proceedings/05/Papers/0763.pdf

[13] Ian Covert and Su-In Lee. 2021. Improving KernelSHAP: Practical Shapley Value Estimation Using Linear Regression. In *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, April 13-15, 2021, Virtual Event (Proceedings of Machine Learning Research, Vol. 130)*, Arindam Banerjee and Kenji Fukumizu (Eds.). PMLR, 3457–3465. http://proceedings.mlr.press/v130/covert21a.html

[14] Susan B. Davidson, Daniel Deutch, Nave Frost, Benny Kimelfeld, Omer Koren, and Mikaël Monet. 2022. ShapGraph: An Holistic View of Explanations through Provenance Graphs and Shapley Values. In *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, Zachary G. Ives, Angela Bonifati, and Amr El Abbadi (Eds.). ACM, 2373–2376. https://doi.org/10.1145/3514221.3520172

[15] Xiaotie Deng and Christos H. Papadimitriou. 1994. On the Complexity of Cooperative Solution Concepts. *Math. Oper. Res.* 19, 2 (1994), 257–266. https://doi.org/10.1287/MOOR.19.2.257

[16] Daniel Deutch, Nave Frost, Benny Kimelfeld, and Mikaël Monet. 2022. Computing the Shapley Value of Facts in Query Answering. In *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, Zachary G. Ives, Angela Bonifati, and Amr El Abbadi (Eds.). ACM, 1570–1583. https://doi.org/10.1145/3514221.3517912

[17] Zhenan Fan, Huang Fang, Xinglu Wang, Zirui Zhou, Jian Pei, Michael Friedlander, and Yong Zhang. 2024. Fair and Efficient Contribution Valuation for Vertical Federated Learning. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=sLQb8q0sUi

[18] Zhenan Fan, Huang Fang, Zirui Zhou, Jian Pei, Michael P. Friedlander, Changxin Liu, and Yong Zhang. 2022. Improving Fairness for Data Valuation in Horizontal Federated Learning. In *38th IEEE International Conference on Data Engineering, ICDE 2022, Kuala Lumpur, Malaysia, May 9-12, 2022*. IEEE, 2440–2453. https://doi.org/10.1109/ICDE53745.2022.00228

[19] Eitan Farchi, Ramasuri Narayanam, and Lokesh Nagalapatti. 2021. Ranking Data Slices for ML Model Validation: A Shapley Value Approach. In *37th IEEE International Conference on Data Engineering, ICDE 2021, Chania, Greece, April 19-22, 2021*. IEEE, 1937–1942. https://doi.org/10.1109/ICDE51399.2021.00180

[20] Raul Castro Fernandez. 2022. Protecting Data Markets from Strategic Buyers. In *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, Zachary G. Ives, Angela Bonifati, and Amr El Abbadi (Eds.). ACM, 1755–1769. https://doi.org/10.1145/3514221.3517855

[21] R. A. Fisher. 1988. Iris. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C56C76.

[22] Yihan Geng, Kunyu Wang, Ziqi Liu, Michael Yu, and Jeffrey Xu Yu. 2023. Influence Maximization Revisited. In *Databases Theory and Applications - 34th Australasian Database Conference, ADC 2023, Melbourne, VIC, Australia, November 1-3, 2023, Proceedings (Lecture Notes in Computer Science, Vol. 14386)*, Zhifeng Bao, Renata Borovica-Gajic, Ruihong Qiu, Farhana Murtaza Choudhury, and Zhengyi Yang (Eds.). Springer, 356–370. https://doi.org/10.1007/978-3-031-47843-7_25

[23] Amirata Ghorbani, Michael P. Kim, and James Zou. 2020. A Distributional Framework For Data Valuation. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event (Proceedings of Machine Learning Research, Vol. 119)*. PMLR, 3535–3544. http://proceedings.mlr.press/v119/ghorbani20a.html

[24] Amirata Ghorbani and James Y. Zou. 2019. Data Shapley: Equitable Valuation of Data for Machine Learning. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR,

2242–2251. http://proceedings.mlr.press/v97/ghorbani19c.html

[25] Stefan Grafberger, Shubha Guha, Paul Groth, and Sebastian Schelter. 2023. MLWHATIF: What If You Could Stop Re-Implementing Your Machine Learning Pipeline Analyses Over and Over? *Proc. VLDB Endow.* 16, 12 (2023), 4002–4005. https://doi.org/10.14778/3611540.3611606

[26] Ben Halstead, Yun Sing Koh, Patricia Riddle, Mykola Pechenizkiy, Albert Bifet, and Russel Pears. 2021. Fingerprinting Concepts in Data Streams with Supervised and Unsupervised Meta-Information. In *37th IEEE International Conference on Data Engineering, ICDE 2021, Chania, Greece, April 19-22, 2021*. IEEE, 1056–1067. https://doi.org/10.1109/ICDE51399.2021.00096

[27] W. Hoeffding. 1963. Probability Inequalities for Sums of Bounded Random Variables. *J. Amer. Statist. Assoc.* 58, 301 (1963), 13–30.

[28] Wassily Hoeffding. 1994. Probability inequalities for sums of bounded random variables. *The collected works of Wassily Hoeffding* (1994), 409–426.

[29] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gürel, Bo Li, Ce Zhang, Costas J. Spanos, and Dawn Song. 2019. Efficient Task-Specific Data Valuation for Nearest Neighbor Algorithms. *Proc. VLDB Endow.* 12, 11 (2019), 1610–1623. https://doi.org/10.14778/3342263.3342637

[30] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J. Spanos. 2019. Towards Efficient Data Valuation Based on the Shapley Value. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan (Proceedings of Machine Learning Research, Vol. 89)*, Kamalika Chaudhuri and Masashi Sugiyama (Eds.). PMLR, 1167–1176. http://proceedings.mlr.press/v89/jia19a.html

[31] Ahmet Kara, Dan Olteanu, and Dan Suciu. 2024. From Shapley Value to Model Counting and Back. *PODS* 2, 2 (2024), 79. https://doi.org/10.1145/3651142

[32] Bojan Karlaš, David Dao, Matteo Interlandi, Sebastian Schelter, Wentao Wu, and Ce Zhang. 2024. Data Debugging with Shapley Importance over Machine Learning Pipelines. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=qxGXjWxabq

[33] Pratik Karmakar, Mikaël Monet, Pierre Senellart, and Stéphane Bressan. 2024. Expected Shapley-Like Scores of Boolean functions: Complexity and Applications to Probabilistic Databases. *SIGMOD* 2, 2 (2024), 92. https://doi.org/10.1145/3651593

[34] Patrick Kolpaczki, Viktor Bengs, Maximilian Muschalik, and Eyke Hüllermeier. 2024. Approximating the Shapley Value without Marginal Contributions. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (Eds.). AAAI Press, 13246–13255. https://doi.org/10.1609/AAAI.V38I12.29225

[35] Feifei Li. 2023. Modernization of Databases in the Cloud Era: Building Databases that Run Like Legos. *Proc. VLDB Endow.* 16, 12 (2023), 4140–4151. https://doi.org/10.14778/3611540.3611639

[36] Jinyang Li, Yuval Moskovitch, and H. V. Jagadish. 2023. Detection of Groups with Biased Representation in Ranking. In *39th IEEE International Conference on Data Engineering, ICDE 2023, Anaheim, CA, USA, April 3-7, 2023*. IEEE, 2167–2179. https://doi.org/10.1109/ICDE55515.2023.00168

[37] Weida Li and Yaoliang Yu. 2024. Faster Approximation of Probabilistic and Distributional Values via Least Squares. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net. https://openreview.net/forum?id=lvSMIsztka

[38] Ye Li, Jian Tan, Bin Wu, Xiao He, and Feifei Li. 2023. ShapleyIQ: Influence Quantification by Shapley Values for Performance Debugging of Microservices. In *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 4, ASPLOS 2023, Vancouver, BC, Canada, March 25-29, 2023*, Tor M. Aamodt, Michael M. Swift, and Natalie D. Enright Jerger (Eds.). ACM, 287–323. https://doi.org/10.1145/3623278.3624771

[39] Stephen C Littlechild and Guillermo Owen. 1973. A simple expression for the Shapley value in a special case. *Management Science* 20, 3 (1973), 370–372.

[40] Stephen C Littlechild and GF Thompson. 1977. Aircraft landing fees: a game theory approach. *The Bell Journal of Economics* (1977), 186–204.

[41] Jinfei Liu, Qiongqiong Lin, Jiayao Zhang, Kui Ren, Jian Lou, Junxu Liu, Li Xiong, Jian Pei, and Jimeng Sun. 2021. Demonstration of Dealer: An End-to-End Model Marketplace with Differential Privacy. *Proc. VLDB Endow.* 14, 12 (2021), 2747–2750. https://doi.org/10.14778/3476311.3476335

[42] Jinfei Liu, Jian Lou, Junxu Liu, Li Xiong, Jian Pei, and Jimeng Sun. 2021. Dealer: An End-to-End Model Marketplace with Differential Privacy. *Proc. VLDB Endow.* 14, 6 (2021), 957–969. https://doi.org/10.14778/3447689.3447700

[43] Ester Livshits, Leopoldo E. Bertossi, Benny Kimelfeld, and Moshe Sebag. 2020. The Shapley Value of Tuples in Query Answering. In *23rd International Conference on Database Theory, ICDT 2020, March 30-April 2, 2020, Copenhagen,*

Denmark (LIPIcs, Vol. 155), Carsten Lutz and Jean Christoph Jung (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 20:1–20:19. https://doi.org/10.4230/LIPICS.ICDT.2020.20

[44] Scott M. Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 4765–4774. https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html

[45] Xuan Luo and Jian Pei. 2024. Applications and Computation of the Shapley Value in Databases and Machine Learning. In *Companion of the 2024 International Conference on Management of Data, SIGMOD/PODS 2024, Santiago AA, Chile, June 9-15, 2024*, Pablo Barceló, Nayat Sánchez Pi, Alexandra Meliou, and S. Sudarshan (Eds.). ACM, 630–635. https://doi.org/10.1145/3626246.3654680

[46] Xuan Luo, Jian Pei, Zicun Cong, and Cheng Xu. 2022. On Shapley Value in Data Assemblage Under Independent Utility. *Proc. VLDB Endow.* 15, 11 (2022), 2761–2773. https://doi.org/10.14778/3551793.3551829

[47] Xuan Luo, Jian Pei, Cheng Xu, Wenjie Zhang, and Jianliang Xu. 2024. Fast Shapley Value Computation in Data Assemblage Tasks as Cooperative Simple Games. *SIGMOD* 2, 1 (2024), 56:1–56:28. https://doi.org/10.1145/3639311

[48] Shuaicheng Ma, Yang Cao, and Li Xiong. 2021. Transparent Contribution Evaluation for Secure Federated Learning on Blockchain. In *37th IEEE International Conference on Data Engineering Workshops, ICDE Workshops 2021, Chania, Greece, April 19-22, 2021*. IEEE, 88–91. https://doi.org/10.1109/ICDEW53142.2021.00023

[49] Sasan Maleki. 2015. *Addressing the computational issues of the Shapley value with applications in the smart grid.* Ph. D. Dissertation. University of Southampton, UK. http://eprints.soton.ac.uk/383963/

[50] Irwin Mann and Lloyd S Shapley. 1960. *Values of large games, IV: Evaluating the electoral college by Montecarlo techniques.* Rand Corporation.

[51] Rory Mitchell, Joshua Cooper, Eibe Frank, and Geoffrey Holmes. 2022. Sampling Permutations for Shapley Value Estimation. *J. Mach. Learn. Res.* 23 (2022), 43:1–43:46. http://jmlr.org/papers/v23/21-0439.html

[52] Nikolaos Myrtakis, Ioannis Tsamardinos, and Vassilis Christophides. 2021. PROTEUS: Predictive Explanation of Anomalies. In *37th IEEE International Conference on Data Engineering, ICDE 2021, Chania, Greece, April 19-22, 2021*. IEEE, 1967–1972. https://doi.org/10.1109/ICDE51399.2021.00185

[53] Sellers Tracy Talbot Simon Cawthorn Andrew Nash, Warwick and Wes Ford. 1994. Abalone. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C55C7W.

[54] Art B. Owen. 2013. *Monte Carlo theory, methods and examples.* https://artowen.su.domains/mc/.

[55] Guillermo Owen. 2013. *Game theory.* Emerald Group Publishing.

[56] Romila Pradhan, Aditya Lahiri, Sainyam Galhotra, and Babak Salimi. 2022. Explainable AI: Foundations, Applications, Opportunities for Data Management Research. In *38th IEEE International Conference on Data Engineering, ICDE 2022, Kuala Lumpur, Malaysia, May 9-12, 2022*. IEEE, 3209–3212. https://doi.org/10.1109/ICDE53745.2022.00300

[57] Alon Reshef, Benny Kimelfeld, and Ester Livshits. 2020. The Impact of Negation on the Complexity of the Shapley Value in Conjunctive Queries. In *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020*, Dan Suciu, Yufei Tao, and Zhewei Wei (Eds.). ACM, 285–297. https://doi.org/10.1145/3375395.3387664

[58] Alvin E Roth. 1988. *The Shapley value: essays in honor of Lloyd S. Shapley.* Cambridge University Press.

[59] Sebastian Schelter, Stefan Grafberger, Shubha Guha, Bojan Karlas, and Ce Zhang. 2023. Proactively Screening Machine Learning Pipelines with ARGUSEYES. In *Companion of the 2023 International Conference on Management of Data, SIGMOD/PODS 2023, Seattle, WA, USA, June 18-23, 2023*, Sudipto Das, Ippokratis Pandis, K. Selçuk Candan, and Sihem Amer-Yahia (Eds.). ACM, 91–94. https://doi.org/10.1145/3555041.3589682

[60] Lloyd S. Shapley. 1953. A value for n-person games. (1953).

[61] Tianshu Song, Yongxin Tong, and Shuyue Wei. 2019. Profit Allocation for Federated Learning. In *2019 IEEE International Conference on Big Data (IEEE BigData), Los Angeles, CA, USA, December 9-12, 2019*, Chaitanya K. Baru, Jun Huan, Latifur Khan, Xiaohua Hu, Ronay Ak, Yuanyuan Tian, Roger S. Barga, Carlo Zaniolo, Kisung Lee, and Yanfang (Fanny) Ye (Eds.). IEEE, 2577–2586. https://doi.org/10.1109/BIGDATA47090.2019.9006327

[62] Qiheng Sun, Xiang Li, Jiayao Zhang, Li Xiong, Weiran Liu, Jinfei Liu, Zhan Qin, and Kui Ren. 2023. ShapleyFL: Robust Federated Learning Based on Shapley Value. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, Ambuj K. Singh, Yizhou Sun, Leman Akoglu, Dimitrios Gunopulos, Xifeng Yan, Ravi Kumar, Fatma Ozcan, and Jieping Ye (Eds.). ACM, 2096–2108. https://doi.org/10.1145/3580305.3599500

[63] Sofiane Touati, Mohammed Said Radjef, and Lakhdar Sais. 2021. A Bayesian Monte Carlo method for computing the Shapley value: Application to weighted voting and bin packing games. *Comput. Oper. Res.* 125 (2021), 105094. https://doi.org/10.1016/J.COR.2020.105094

[64] Guan Wang. 2019. Interpret Federated Learning with Shapley Values. *CoRR* abs/1905.04519 (2019). arXiv:1905.04519 http://arxiv.org/abs/1905.04519

[65] Junhao Wang, Lan Zhang, Anran Li, Xuanke You, and Haoran Cheng. 2022. Efficient Participant Contribution Evaluation for Horizontal and Vertical Federated Learning. In *38th IEEE International Conference on Data Engineering, ICDE 2022, Kuala Lumpur, Malaysia, May 9-12, 2022*. IEEE, 911–923. https://doi.org/10.1109/ICDE53745.2022.00073

[66] Tingting Wang, Shixun Huang, Zhifeng Bao, J. Shane Culpepper, Volkan Dedeoglu, and Reza Arablouei. 2024. Optimizing Data Acquisition to Enhance Machine Learning Performance. *Proc. VLDB Endow.* 17, 6 (2024), 1310–1323. https://www.vldb.org/pvldb/vol17/p1310-bao.pdf

[67] Lauren Watson, Zeno Kujawa, Rayna Andreeva, Hao-Tsung Yang, Tariq Elahi, and Rik Sarkar. 2023. Accelerated Shapley Value Approximation for Data Evaluation. *CoRR* abs/2311.05346 (2023). https://doi.org/10.48550/ARXIV.2311.05346 arXiv:2311.05346

[68] WIlliam Wolberg. 1992. Breast Cancer Wisconsin (Original). UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5HP4Z.

[69] Mengmeng Wu, Ruoxi Jia, Changle Lin, Wei Huang, and Xiangyu Chang. 2023. Variance reduced Shapley value estimation for trustworthy data valuation. *Comput. Oper. Res.* 159 (2023), 106305. https://doi.org/10.1016/J.COR.2023.106305

[70] Haocheng Xia, Xiang Li, Junyuan Pang, Jinfei Liu, Kui Ren, and Li Xiong. 2024. P-Shapley: Shapley Values on Probabilistic Classifiers. *Proc. VLDB Endow.* 17, 7 (2024), 1737–1750. https://www.vldb.org/pvldb/vol17/p1737-liu.pdf

[71] Haocheng Xia, Jinfei Liu, Jian Lou, Zhan Qin, Kui Ren, Yang Cao, and Li Xiong. 2023. Equitable Data Valuation Meets the Right to Be Forgotten in Model Markets. *Proc. VLDB Endow.* 16, 11 (2023), 3349–3362. https://doi.org/10.14778/3611479.3611531

[72] Jiayao Zhang, Qiheng Sun, Jinfei Liu, Li Xiong, Jian Pei, and Kui Ren. 2023. Efficient Sampling Approaches to Shapley Value Approximation. *SIGMOD* 1, 1 (2023), 48:1–48:24. https://doi.org/10.1145/3588728

[73] Jiayao Zhang, Haocheng Xia, Qiheng Sun, Jinfei Liu, Li Xiong, Jian Pei, and Kui Ren. 2023. Dynamic Shapley Value Computation. In *39th IEEE International Conference on Data Engineering, ICDE 2023, Anaheim, CA, USA, April 3-7, 2023*. IEEE, 639–652. https://doi.org/10.1109/ICDE55515.2023.00055

[74] Xinyi Zhang, Zhuo Chang, Yang Li, Hong Wu, Jian Tan, Feifei Li, and Bin Cui. 2022. Facilitating Database Tuning with Hyper-Parameter Optimization: A Comprehensive Experimental Evaluation. *Proc. VLDB Endow.* 15, 9 (2022), 1808–1821. https://doi.org/10.14778/3538598.3538604

[75] Shuyuan Zheng, Yang Cao, and Masatoshi Yoshikawa. 2023. Secure Shapley Value for Cross-Silo Federated Learning. *Proc. VLDB Endow.* 16, 7 (2023), 1657–1670. https://doi.org/10.14778/3587136.3587141

[76] Guanghui Zhu, Wenjie Wang, Zhuoer Xu, Feng Cheng, Mengchuan Qiu, Chunfeng Yuan, and Yihua Huang. 2022. PSP: Progressive Space Pruning for Efficient Graph Neural Architecture Search. In *38th IEEE International Conference on Data Engineering, ICDE 2022, Kuala Lumpur, Malaysia, May 9-12, 2022*. IEEE, 2168–2181. https://doi.org/10.1109/ICDE53745.2022.00208

[77] Yuqing Zhu, Jing Tang, Xueyan Tang, and Lei Chen. 2021. Analysis of Influence Contribution in Social Advertising. *Proc. VLDB Endow.* 15, 2 (2021), 348–360. https://doi.org/10.14778/3489496.3489514

A Algorithm

B Proofs

B.1 Proof of Theorem 4.3

Proof. We solve the least-squares problem by transforming it into a standard form $\min ||Ax - b||^2$.

In this case, we let $x = \begin{pmatrix} \mathcal{SV}_1 \\ \vdots \\ \mathcal{SV}_n \end{pmatrix}$ and $b = \begin{pmatrix} \Delta\mathcal{SV}_{1,2} \\ \Delta\mathcal{SV}_{1,3} \\ \vdots \\ \Delta\mathcal{SV}_{n-1,n} \end{pmatrix}$.

We could find that x has dimension n and b has dimension $n(n-1)/2$. Then we could obtain that A is a matrix with size $n(n-1)/2 \times n$, and

$$A^T = \begin{pmatrix} 1 & 1 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ -1 & 0 & \cdots & 0 & 1 & 1 & \cdots & 0 \\ 0 & -1 & \cdots & 0 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 & 0 & 0 & \cdots & -1 \end{pmatrix}.$$

Then $||Ax - b||^2$ is the same as the original objective function. We know that the analytic solution of the least-squares problem satisfies the linear equations system $A^T Ax = A^T b$. Since $\sum_{i=1}^{n} \mathcal{SV}_i = \mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset)$, we have $\mathcal{SV}_i = \frac{1}{n}\sum_{j=1}^{n} \Delta\mathcal{SV}_{i,j} + \frac{1}{n}[\mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset)]$. □

Algorithm 6: GreddyFill($\mathcal{N}$)

for $k = 1$ *to* $\lfloor \frac{n}{2} \rfloor$ **do**
  $\mathcal{L} \leftarrow \mathcal{N}$;
  while $\exists\, m^k_{i,j} = 0$ **do**
    $\mathcal{S} \leftarrow \emptyset$, $l \leftarrow 1$;
    while $|\mathcal{S}| < k$ **do**
      if $\mathcal{L} = \emptyset$ **then**
        $\mathcal{L} = \mathcal{N}$;
      Seek $\mathcal{J}$ containing all the elements s.t. $m^k_{l,j} = 0$;
      if $\mathcal{J} = \emptyset$ **then**
        l+ = 1;
        continue;
      else
        if $z_l \in \mathcal{L}$ **then**
          $\mathcal{S} = \mathcal{S} \cup \{z_l\}$, $\mathcal{L} = \mathcal{L} \setminus \mathcal{J}$;
        l+ = k;
    $u \leftarrow \mathcal{U}(\mathcal{S})$, $nu \leftarrow \mathcal{U}(\mathcal{N} \setminus \mathcal{S})$;
    for $z_i \in \mathcal{S}$ **do**
      for $z_j \notin \mathcal{S}$ **do**
        $\widehat{\mathcal{U}^k_{\pi^t(i),\pi^t(j)}}$ += u, $m^k_{i,j}$ += 1;
        $\widehat{\mathcal{U}^{n-k}_{\pi^t(j),\pi^t(i)}}$ += nu,$m^{n-k}_{j,i}$ += 1;
        c+ = 2;

B.2 Proof of Theorem 5.1

Proof. According to Equation 1, we have (denote $\mathcal{N} \setminus \{z_i\}$ by $\mathcal{N}_{\setminus i}$ and $\mathcal{S} \cup \{z_i\}$ by $\mathcal{S}_{\cup i}$)

$$\begin{aligned}
\mathcal{SV}_i - \mathcal{SV}_j =& \frac{1}{n}\Big[\sum_{\mathcal{S}\subseteq\mathcal{N}_{\setminus i}} \frac{\mathcal{U}(\mathcal{S}_{\cup i}) - \mathcal{U}(\mathcal{S})}{\binom{n-1}{|\mathcal{S}|}} - \sum_{\mathcal{S}\subseteq\mathcal{N}_{\setminus j}} \frac{\mathcal{U}(\mathcal{S}_{\cup j}) - \mathcal{U}(\mathcal{S})}{\binom{n-1}{|\mathcal{S}|}}\Big] \\
=& \frac{1}{n} \sum_{\mathcal{S}\subseteq\mathcal{N}_{\setminus i,j}} \frac{\mathcal{U}(\mathcal{S}_{\cup i}) - \mathcal{U}(\mathcal{S}_{\cup j})}{\binom{n-1}{|\mathcal{S}|}} + \frac{\mathcal{U}(\mathcal{S}_{\cup i}) - \mathcal{U}(\mathcal{S}_{\cup j})}{\binom{n-1}{|\mathcal{S}|+1}} \\
=& \frac{1}{n-1} \sum_{\mathcal{S}\subseteq\mathcal{N}_{\setminus i,j}} \frac{\mathcal{U}(\mathcal{S}_{\cup i}) - \mathcal{U}(\mathcal{S}_{\cup j})}{\binom{n-2}{|\mathcal{S}|}} = \frac{1}{n-1}\sum_{z_i\in\mathcal{S}} \frac{\mathcal{U}(\mathcal{S})}{\binom{n-2}{|\mathcal{S}|-1}} - \frac{1}{n-1}\sum_{z_j\in\mathcal{S}} \frac{\mathcal{U}(\mathcal{S})}{\binom{n-2}{|\mathcal{S}|-1}}.
\end{aligned}$$

□

B.3 Proof of Theorem 5.3

Proof. We have $\mathcal{U}_{i,j} = \frac{1}{n-1}\sum_{z_i\in\mathcal{S}, z_j\notin\mathcal{S}} \frac{\mathcal{U}(\mathcal{S})}{\binom{n-2}{|\mathcal{S}|-1}}$. The probability of a specific coalition $\mathcal{S}$ being selected is proportional to $\frac{1}{|\mathcal{S}|(n-|\mathcal{S}|)\binom{n}{|\mathcal{S}|}} = \frac{|\mathcal{S}|!(n-|\mathcal{S}|)!}{|\mathcal{S}|(n-|\mathcal{S}|)n!} = \frac{(|\mathcal{S}|-1)!(n-|\mathcal{S}|-1)!}{n!} = \frac{1}{n(n-1)\binom{n-2}{|\mathcal{S}|-1}} \propto \frac{1}{\binom{n-2}{|\mathcal{S}|-1}}$. Therefore, $\widehat{\mathcal{U}_{i,j}}/m_{i,j}$ is an unbiased estimator of $\mathcal{U}_{i,j}$. According to Theorem 5.1, $\Delta\mathcal{SV}_{i,j} = \mathcal{U}_{i,j} - \mathcal{U}_{j,i}$. Therefore, $\widehat{\Delta\mathcal{SV}_{i,j}} = \widehat{\mathcal{U}_{i,j}}/m_{i,j} - \widehat{\mathcal{U}_{j,i}}/m_{j,i}$ is an unbiased estimator of $\Delta\mathcal{SV}_{i,j}$. □

B.4 Proof of Theorem 5.8

Proof. The probability that a coalition $\mathcal{S}$ in $\mathfrak{S}^k$ belongs to $\mathfrak{S}^k_{i\setminus j}$ is the probability that z_i belongs to $\mathcal{S}$ and z_j does not, i.e., $\frac{k(n-k)}{n(n-1)}$. Thus, it is easy to see that with m_k samples observed, the expected sample size of $\mathfrak{S}^k_{i\setminus j}$ is

$$\mathrm{E}[m_{i,j,k}] = \frac{k(n-k)}{n(n-1)} m_k.$$

Therefore, we have

$$\mathrm{E}\Big[\sum_{1\le i<j\le n} \mathrm{Var}(\widehat{\Delta\mathcal{SV}_{i,j}})\Big] = \frac{1}{n^2(n-1)^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n-1}\frac{\sigma^2_{i,j,k}}{m_{i,j,k}} = \frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n-1}\frac{\sigma^2_{i,j,k}}{k(n-k)m_k}.$$

□

B.5 Proof of Theorem 5.9

Proof. According to Theorem 5.8 and using the Cauchy-Schwarz inequality, we can get

$$\mathrm{E}[\sum_{1\le i<j\le n}\mathrm{Var}(\widehat{\Delta\mathcal{SV}_{i,j}})]\sum_{k=1}^{n-1}m_k \ge (\sum_{k=1}^{n-1}\sqrt{\frac{n(n-1)}{k(n-k)}\sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_{i,j,k}^2})^2.$$

According to the equality condition of the Cauchy-Schwarz inequality, we have

$$m_k \propto \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\sigma_{i,j,k}^2}{k(n-k)}}.$$

□

B.6 Proof of Theorem 6.1

Proof. Note that all $\widehat{\Delta\mathcal{SV}_{i,j}}$ have the same expectation of sample sizes. Assume it is m, then $\mathrm{E}[\mathrm{Var}(\widehat{\Delta\mathcal{SV}_{i,j}})] = \frac{1}{m}\sigma_{i,j}^2$. Thus we only need to consider

$$\mathcal{SV}_1^P = \frac{1}{n}\sum_{j=1}^{n}\Delta\mathcal{SV}_{1,j} + \frac{1}{n}(\mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset)), \mathcal{SV}_i^P = \mathcal{SV}_1 - \Delta\mathcal{SV}_{1,i} \quad (i\neq 1), \mathcal{SV}_i^F = \frac{1}{n}\sum_{j=1}^{n}\Delta\mathcal{SV}_{i,j} + \frac{1}{n}(\mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset))$$

as discrete random variables, and prove that when $n > 5$,

$$\sum_{i=1}^{n}\mathrm{Var}(\mathcal{SV}_i^P) > \sum_{i=1}^{n}\mathrm{Var}(\mathcal{SV}_i^F).$$

Since

$$\mathrm{Var}(\mathcal{SV}_1^P) = \mathrm{Var}\left(\frac{1}{n}\sum_{j=1}^{n}\Delta\mathcal{SV}_{1,j}\right) = \frac{1}{n^2}\sum_{j=1}^{n}\sigma_{1,j}^2,$$

$$\begin{aligned}\mathrm{Var}(\mathcal{SV}_i^P) &= \mathrm{Var}\left(\frac{1}{n}\sum_{j\neq i}\Delta\mathcal{SV}_{1,j} - \frac{n-1}{n}\Delta\mathcal{SV}_{1,i}\right)\\ &= \frac{1}{n^2}\sum_{j\neq i}\sigma_{1,j}^2 + \frac{(n-1)^2}{n^2}\sigma_{1,i}^2 \quad (i\neq 1),\end{aligned}$$

we have

$$\sum_{i=1}^{n}\mathrm{Var}(\mathcal{SV}_i^P) = \left(\frac{(n-1)^2}{n^2} + \frac{(n-1)}{n^2}\right)\sum_{j=1}^{n}\sigma_{1,j}^2 = \frac{(n-1)}{n}\sum_{j=1}^{n}\sigma_{1,j}^2.$$

On the other hand,

$$\mathrm{Var}(\mathcal{SV}_i^F) = \mathrm{Var}\left(\frac{1}{n}\sum_{j=1}^{n}\Delta\mathcal{SV}_{i,j}\right) = \frac{1}{n^2}\sum_{j=1}^{n}\mathrm{Var}(\Delta\mathcal{SV}_{i,j}) = \frac{1}{n^2}\sum_{j=1}^{n}\sigma_{i,j}^2.$$

Then

$$\sum_{i=1}^{n}\mathrm{Var}(\mathcal{SV}_i^F) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_{i,j}^2 \le \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}2(\sigma_{1,i}^2+\sigma_{1,j}^2) = \frac{4}{n}\sum_{j=1}^{n}\sigma_{1,j}^2 \le \frac{n-1}{n}\sum_{j=1}^{n}\sigma_{1,j}^2 = \sum_{i=1}^{n}\mathrm{Var}(\mathcal{SV}_i^P).$$

□

B.7 Proof of Theorem 6.2

Proof. **Algorithm 1**: According to Theorem 4.3, we have

$$\begin{aligned}\mathrm{Var}(\widehat{\mathcal{SV}_i}) &= \mathrm{Var}\left(\frac{1}{n}\sum_{j=1}^{n}\widehat{\Delta\mathcal{SV}_{i,j}} + \frac{1}{n}(\mathcal{U}(\mathcal{N}) - \mathcal{U}(\emptyset))\right) = \frac{1}{n^2}\sum_{j=1}^{n}\mathrm{Var}(\widehat{\Delta\mathcal{SV}_{i,j}}) + \frac{2}{n^2}\sum_{j=1}^{n}\sum_{l=1}^{n}\mathrm{Cov}(\widehat{\Delta\mathcal{SV}_{i,j}}, \widehat{\Delta\mathcal{SV}_{i,l}})\\ &= \frac{1}{n^2}\sum_{j=1}^{n}\frac{\sigma_{i,j}^2}{m_{i,j}} + \frac{2}{n^2}\sum_{j=1}^{n}\sum_{l=1}^{n}\frac{\sigma_{i,j,l}^2}{m_{i,j}}.\end{aligned}$$

Similar to the proof of Theorem 5.8, we have

$$\mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i^F})] = \frac{2(n-1)\sum_{k=1}^{n-1}\frac{1}{k}}{nm}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\sigma_{i,j}^2 + \sum_{l=1}^{n}\sigma_{i,j,l}^2\right).$$

Algorithm 2: Similar to the proof of Theorem 5.8 with $m_k = \frac{\frac{1}{k(n-k)}}{\sum_{k=1}^{n-1}\frac{1}{k(n-k)}}$, we can calculate

$$\mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i^F})] = \frac{2(n-1)\sum_{k=1}^{n-1}\frac{1}{k}}{nm}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n-1}\left(\sigma_{i,j,k}^2 + \sum_{l=1}^{n}\sigma_{i,j,l,k}^2\right).$$

□

B.8 Proof of Theorem 6.3

Proof. Following the work of Li and Yu [37], we have $\mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i^U})] = \frac{2(\sum_{k=1}^{n}\frac{1}{k})^2}{m}\sum_{i=1}^{n}\sigma_i^2$, where $\sigma_{i,j}$ is the variance of the random variable over the set $\{\mathcal{U}(\mathcal{S})|z_i \in \mathcal{S}\}$ with $Pr(\mathcal{U} = \mathcal{U}(\mathcal{S})) = \frac{1}{\binom{n-2}{|\mathcal{S}|-1}}$. Then we can calculate $c_{\mathcal{U}} = \lim_{m\to+\infty}\mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i^F})]/\mathrm{E}[\sum_{i=1}^{n}\mathrm{Var}(\widehat{\mathcal{SV}_i^U})]$. Therefore, $c_{\mathcal{U}} \le 1 - 1/n$. Moreover, with the condition $2\mathrm{Cov}(\widehat{\mathcal{SV}_i^U}, \widehat{\mathcal{SV}_j^U}) > c_0(\mathrm{Var}(\widehat{\mathcal{SV}_i^U}) + \mathrm{Var}(\widehat{\mathcal{SV}_j^U}))$, to make $\sum_{i=1}^{n}\widehat{\mathcal{SV}_i^U} = \mathcal{U}(\mathcal{N})$. We have

$$\mathrm{E}[\mathrm{Var}(\widehat{\mathcal{SV}_i^F})] = \frac{(n-1)^2}{n^2}\mathrm{E}[\mathrm{Var}(\widehat{\mathcal{SV}_i^U})] + \frac{1}{n^2}\sum_{j\neq i}\mathrm{E}[\mathrm{Var}(\widehat{\mathcal{SV}_j^U})] - \frac{2(n-1)}{n^2}\sum_{j\neq i}\mathrm{E}[\mathrm{Cov}(\widehat{\mathcal{SV}_i^U}, \widehat{\mathcal{SV}_j^U})].$$

Therefore, we have $\sum_i^n \mathrm{E}[\mathrm{Var}(\widehat{\mathcal{SV}_i^F})] = \frac{n-1}{n}\sum_i^n \mathrm{E}[\mathrm{Var}(\widehat{\mathcal{SV}_i^U})] - \frac{2(n-1)}{n^2}\sum_{1\le i<j\le n}\mathrm{E}[\mathrm{Cov}(\widehat{\mathcal{SV}_i^U}, \widehat{\mathcal{SV}_j^U})] < \frac{(n-1)(1-c_0)}{n}\sum_i^n \mathrm{E}[\mathrm{Var}(\widehat{\mathcal{SV}_i^U})]$, which means $c_{\mathcal{U}} \le (1-1/n)(1-c_0)$. □

B.9 Proof of Theorem 6.4

Proof. We state the case that two algorithms have the sample allocation (i.e., identical m_k). Denote by $CC_{\mathcal{N}}^{i,k} = CC_{\mathcal{N}}(\mathcal{S}), \mathcal{U}^{i,k} = \mathcal{U}(\mathcal{S}), (|\mathcal{S}| = k, z_i \in \mathcal{S})$, and $\sigma_{i,k}^2 = \mathrm{Var}(CC_{\mathcal{N}}^{i,k})$. Following the work of Zhang et al. [72], we have (note that $\sigma_{i,n}^2 = 0$) $\mathrm{Var}(\widehat{\mathcal{SV}_i^C}) = \frac{1}{n^2}\sum_{k=1}^{n-1}\frac{1}{m_{i,k}}\sigma_{i,k}^2$, where $m_{i,k}$ is the number of samples for $CC_{\mathcal{N}}^{i,k}$. Note that ρ_m^C is also the ratio between the coefficient of the coalition $\mathcal{S}$ in the expression of $\widehat{\mathcal{SV}^F}$ and the coefficient of the sample $\mathcal{S}$ in the expression of $\widehat{\mathcal{SV}^C}$, which is getting smaller when m is insufficient. The limitation of this ratio is no more than $1 - 1/n$.

Moreover, since we have the condition $\mathcal{U}(\mathcal{N}\setminus\mathcal{S}_1) \le \mathcal{U}(\mathcal{N}\setminus\mathcal{S}_2)$. Denote by $\mathcal{U}^{i\setminus k} = \mathcal{U}(\mathcal{S}), (|\mathcal{S}| = k, z_i \notin \mathcal{S})$. We have $\mathrm{Cov}(\mathcal{U}^{i,k}, \mathcal{U}^{i\setminus n-k}) < 0$, which means

$$\mathrm{Var}(CC_{\mathcal{N}}^{i,k}) = \mathrm{Var}(\mathcal{U}^{i,k} - \mathcal{U}^{i\setminus n-k}) > \mathrm{Var}(\mathcal{U}^{i,k}) + \mathrm{Var}(\mathcal{U}^{i\setminus n-k}).$$

Therefore, the variances of sampling complementary contributions are higher than the variances of sampling utilities. Then we have $c_C \le 1 - 1/n$. The rest of the proof is the same as Theorem 6.3. □

B.10 Proof of Theorem 6.5

Proof. According to Theorem 6.3, we only need to prove GELS [37] here. Assume $||\mathcal{U}||_\infty \le u_0$ and $\mathcal{U}(\mathcal{S}) \ge 0$. According to Hoeffding's inequality [27], for any $1 \le i \le n$

$$Pr\left(m_i\left|\widehat{\mathcal{SV}_i} - \mathcal{SV}_i\right| \ge \frac{m_i\varepsilon}{\sqrt{n}}\right) \le 2\exp\left(-\frac{2m_i\varepsilon^2}{nu_0^2}\right),$$

where m_i is the number of samples assigned to the player z_i. According to the inclusion-exclusion principle, if $\left(\sum_{i=1}^{n}\left|\widehat{\mathcal{SV}_i} - \mathcal{SV}_i\right|\right)^{\frac{1}{2}} \ge \varepsilon$, then there exists at least one player z_i that satisfies $\left|\widehat{\mathcal{SV}_i} - \mathcal{SV}_i\right| \ge \frac{\varepsilon}{\sqrt{n}}$. Therefore, if $m_i > \frac{nu_0^2}{2\varepsilon^2}\ln\frac{4n}{\delta}$,

$$Pr\left(\left|\widehat{\mathcal{SV}_i} - \mathcal{SV}_i\right| \ge \frac{\varepsilon}{\sqrt{n}}\right) \le \frac{\delta}{2n}, \text{ and } Pr\left(\left(\sum_{i=1}^{n}\left|\widehat{\mathcal{SV}_i} - \mathcal{SV}_i\right|\right)^{\frac{1}{2}} \ge \varepsilon\right) \le \frac{\delta}{2}.$$

Furthermore, note that $m_i \sim B(m, \frac{1}{2})$. Using the Chernoff bound, if $m > 16\ln\frac{2n}{\delta}$, then $Pr(m_i \le \frac{m}{4}) \le \frac{\delta}{2n}$. In summary, if $m > \frac{2nu_0^2}{\varepsilon^2}\ln\frac{4n}{\delta}$, then $Pr\left(\left(\sum_{i=1}^{n}\left|\widehat{\mathcal{SV}_i} - \mathcal{SV}_i\right|\right)^{\frac{1}{2}} \ge \varepsilon\right) \le \delta$. □

B.11 Proof of Theorem 6.6

Proof. The statement of space cost is trivial. In each sample process, Algorithms 4 and 5 update $|\mathcal{S}| - 1$ or $n - |\mathcal{S}|$ elements with a utility sample $\mathcal{U}(\mathcal{S})$, which means the time cost is $O(n + u(n))$. According to Corollary 5.2, the time cost per sample of Algorithms 2. and 2 is $O(\frac{n^2}{\log n} + u(n))$. With each utility sample $\mathcal{U}(\mathcal{S})$, Algorithm 3 updates $|\mathcal{S}|(n - |\mathcal{S}|)$, so the time cost per sample of Algorithm 3 is $O(n^2 + u(n))$, □

Received July 2024; revised September 2024; accepted November 2024