On Least Squares Approximations for Shapley Values and Applications to Interpretable Machine Learning

Tim Pollmann and Jochen Staudacher *

Posted Date: 29 January 2026

doi: 10.20944/preprints202601.2308.v1

Keywords: cooperative game theory; importance sampling; stratified sampling; Shapley value; interpretable machine learning

 doi:10.20944/preprints202601.2308.v1

Article

On Least Squares Approximations for Shapley Values and Applications to Interpretable Machine Learning

Tim Pollmann and Jochen Staudacher *

Fakultät Informatik, Hochschule Kempten, 87435 Kempten, Germany
* Correspondence: jochen.staudacher@hs-kempten.de; Tel.: +49-831-2523-513

Abstract

The Shapley value stands as the predominant point-valued solution concept in cooperative game theory and has recently become a foundational method in interpretable machine learning. In that domain, a prevailing strategy to circumvent the computational intractability of exact Shapley values is to approximate them by reframing their computation as a weighted least squares optimization problem. We investigate an algorithmic framework by Benati et al. (2019), discuss its feasibility for feature attribution and explore a set of methodological and theoretical refinements, including an approach for sample reuse across strata and a relation to Unbiased KernelSHAP. We conclude with an empirical evaluation of the presented algorithms, assessing their performance on several cooperative games including practical problems from interpretable machine learning.

Keywords: cooperative game theory; importance sampling; stratified sampling; Shapley value; interpretable machine learning

1. Introduction

The original KernelSHAP paper by Lundberg and Lee [1] from 2017 was a landmark success for interpretable machine learning. Its primary achievement was making the Shapley value [2] — a theoretically optimal but computationally intractable [3–5] concept from cooperative game theory [6] — practical for explaining machine learning models. Lundberg and Lee [1] discovered that the exact Shapley values for a prediction could be recovered as the solution to a specially weighted least squares regression problem. By sampling a manageable number of coalitional inputs and solving this approximate regression, KernelSHAP provided a single, consistent metric for feature importance. The accompanying open-source shap library SHAP [1] fueled its widespread use, making it a predominant standard for model interpretation, particularly in the model-agnostic setting.

The follow-up work on Unbiased KernelSHAP by Covert and Lee [7] represents a success in methodological rigor. It addresses the problem that the original KernelSHAP algorithm has not been proven to be unbiased and does not provide uncertainty estimates. While not as publicly visible as the initial breakthrough, this refinement solidified SHAP's foundation as a robust, statistically sound tool.

According to Lundberg and Lee [1], KernelSHAP was developed without knowledge of prior articles on cooperative game theory like Charnes et al. [8] and Ruiz et al. [9], who decades earlier had already derived the precise weighting scheme that makes a least-squares regression yield the Shapley value when complete data is available. This outlines a fascinating case of parallel innovation across disciplines. Cooperative game theory, with the Shapley value [2] as its most important solution concept, had long been a mature and applied field providing rigorous solutions to real-world problems of fair division and coalitional analysis, from allocating airport landing fees among airlines based on runway use [10] to measuring the voting power of political blocs in a legislature [11–13]. Yet, the field of machine learning, increasingly reliant on inscrutable black-box models, had not seen its own predictions as precisely such a system — where input features act as cooperating agents whose joint effort produces a model's output. Lundberg and Lee's pivotal contribution [1] was to bridge this

 doi:10.20944/preprints202601.2308.v1

conceptual gap. They recognized that the abstract "game" defined by a model's prediction function was a natural domain for Shapley's theory. By doing so, they inadvertently reinvented a specific computational tool — the weighted least-squares approximation — from the game theory literature, but with a transformative new purpose: not to analyze economic coalitions, but to explain the inner logic of artificial intelligence.

Whereas KernelSHAP [1] and its unbiased variant [7] are very widely used, the methods from Benati et al. [14] — which are equally based on the least squares formulation of the Shapley value — have received comparatively little attention. This article represents, to our knowledge, their first application in explainable artificial intelligence. We adapt the ideas from Benati et al. [14] as general TU game approximation algorithms, terming their weighted sampling strategy LSS (Least Squares Sampling). Regarding their proposal for stratification, we differentiate S-LSS, an algorithm without sample reuse across strata, from SRS-LSS, an algorithm with sample reuse across strata suggested in Benati et al. [14].

The objective of this paper is neither to introduce a novel Shapley value estimator nor to perform a broad comparison of approximation algorithms. Instead, the contribution of this work is a detailed structural, theoretical, and algorithmic analysis of the methods LSS, S-LSS and SRS-LSS from Benati et al. [14]. We compare the variances of the LSS and S-LSS estimators in detail. We point out how sample reuse across strata may introduce non-zero covariance terms between strata for SRS-LSS. We prove that LSS and UKS approximate the same underlying problem, thereby only differing in their respective sampling strategies. Finally, we test how the unbiased Shapley estimators LSS, S-LSS and SRS-LSS perform in comparison to KernelSHAP and UKS for both classical cooperative games and real-world applications from interpretable machine learning.

The remainder of this article is structured as follows. In Section 2, we summarize basic ideas from cooperative game theory, including linear solution concepts and the Shapley value, and introduce the BShap (Baseline Shapley) model for interpretable machine learning. Section 3 briefly reviews Monte Carlo methods, along with stratified sampling and importance sampling for variance reduction. A summary of results on approximating linear solution concepts by means of importance sampling on the coalition space from our previous work [15] is presented in Section 4. The central developments of this paper are detailed in Section 5. We formally introduce the LSS, S-LSS and SRS-LSS estimators, compare variances of LSS and S-LSS, investigate the emergence of non-zero covariance terms between strata for SRS-LSS and clarify the similarities and differences between LSS and UKS. The empirical performance of the algorithms introduced in Section 5 is analyzed in Section 6 using two types of cooperative games and three real-world explainability scenarios, thereby numerically substantiating our previous analytical claims. The paper concludes in Section 7 with a summary and recommendations.

2. Preliminaries on Cooperative Game Theory, the Shapley Value and the BShap Model for Interpretable Machine Learning

Following the expositions in Chakravarty et al. [6], Peters [16], and Benati et al. [14], this section offers a concise introduction to transferable utility (TU) cooperative games and point-valued solution concepts such as the Shapley value. Additionally, we briefly describe two representative TU games — the airport game and the weighted voting game — and the model from interpretable machine learning which we utilize in our analysis and our numerical experiments in Section 6.

2.1. Transferable Utility Games and Their Characteristic Functions

We investigate *transferable utility games (TU games)*, cooperative games where a coalition's earnings are expressed as a scalar [6,16]. This implies that the utility can be distributed without restriction among the players.

Following [6,16], a TU game is formally defined as a pair (N, v). Here, $N = \{1, \ldots, n\}$ is the player set and $v : 2^N \to \mathbb{R}$ is the *characteristic function*, which assigns a real value to each coalition $S \subseteq N$ representing its total utility from cooperation. We adopt the standard normalization $v(\emptyset) = 0$.

 doi:10.20944/preprints202601.2308.v1

It is frequently convenient to represent a coalition S by an indicator vector $\mathbf{z} \in \{0,1\}^n$. We define $\mathbf{z}(S) = [\mathbb{1}_{i \in S}]_{i=1}^n$ and its inverse $S(\mathbf{z}) = \{i \in N \mid z_i = 1\}$. This mapping allows functions and operators defined on 2^N or $\{0,1\}^n$ to be used interchangeably; for instance, $v(S)$ and $v(\mathbf{z})$ denote the same value, and $|\mathbf{z}| = |S(\mathbf{z})|$.

2.2. Linear Solution Concepts for TU Games

Let $G(N)$ denote the set of all TU games on player set N. A point-valued solution concept is a function $\boldsymbol{\alpha} : G(N) \to \mathbb{R}^n$ that returns a vector $\boldsymbol{\alpha}(N,v) \in \mathbb{R}^n$ for each game $(N,v) \in G(N)$. Each entry α_i quantifies the value assigned to player i. These concepts are employed to evaluate individual influence or to distribute a coalition's payoff $v(S)$ among its members $S \subseteq N$.

A key property of a point-valued solution concept $\boldsymbol{\alpha}$ is linearity [14]. This requires that $\boldsymbol{\alpha}$ can be written as

$$\boldsymbol{\alpha}(N,v) = \sum_{S \subseteq N} \boldsymbol{w}(S) \odot \boldsymbol{\nu}(S,v), \tag{1}$$

with $\odot$ denoting the Hadamard product of vectors, $\boldsymbol{w}(S) = [w_1(S), \ldots, w_n(S)]^\top$ standing for weights depending only on S and the player, and $\boldsymbol{\nu}(S,v) = [\nu_1(S,v), \ldots, \nu_n(S,v)]^\top$ denoting values depending upon S, v, and the player.

2.3. The Shapley Value

The Shapley value [2] represents the most prominent solution concept in cooperative game theory. In recent years, it has been extensively adopted in machine learning and explainable artificial intelligence [1,17–19]. Formally, for any player $i \in N$, the Shapley value is the expected marginal contribution $\Delta_i(S,v) = v(S \cup i) - v(S)$, taken over all sets S of players that precede i in a uniformly random permutation π of N, i.e.,

$$\begin{aligned} \phi_i(N,v) &= \mathbb{E}_{\pi \sim \mathcal{U}(\Pi(N))}\left[\Delta_i\left(\mathrm{Pre}_i(\pi), v\right)\right] && (2)\\ &= \frac{1}{n!} \sum_{\pi \in \Pi(N)} \Delta_i\left(\mathrm{Pre}_i(\pi), v\right) \\ &= \sum_{\substack{S \subseteq N \\ i \notin S}} \frac{|S|!\,(n-|S|-1)!}{n!}\, \Delta_i(S,v), && (3) \end{aligned}$$

where $\mathcal{U}$ stands for a uniform distribution, $\Pi(N)$ denotes the set of all permutations of N, and $\mathrm{Pre}_i(\pi)$ gives the set of players preceding i in a permutation $\pi \in \Pi(N)$.

Let $\boldsymbol{\phi} = [\phi_1, \ldots, \phi_n]^\top$ be the vector of Shapley values, and let $\hat{\boldsymbol{\phi}}$ represent any approximation, with $\hat{\phi}_i$ its i-th entry. Where no ambiguity arises, we use ϕ_i to mean $\phi_i(N,v)$.

It follows directly from (3) that the Shapley value belongs to the class of linear solution concepts introduced in Subsection 2.2. The coefficients $\boldsymbol{w}(S)$ are specified by $\nu_i(S,v) = \Delta_i(S,v)$ and

$$w_i(S) = \begin{cases} \frac{|S|!\,(n-|S|-1)!}{n!} & \text{if } i \notin S \\ 0 & \text{if } i \in S. \end{cases} \tag{4}$$

Note that the Shapley value exhibits *symmetry*, i.e., two players that contribute equally to each coalition have the same Shapley value meaning there holds

$$\text{if } \forall S \subseteq N \setminus \{i,j\},\ v(S \cup \{i\}) = v(S \cup \{j\}), \text{ then } \phi_i(N,v) = \phi_j(N,v),$$

for any game (N,v) and any players $i,j \in N$ with $i \neq j$. It also exhibits *efficiency*, i.e., for all games (N,v) there holds $\sum_{i \in N} \phi_i(N,v) = v(N)$.

 doi:10.20944/preprints202601.2308.v1

2.4. Airport Games

Airport games, proposed by Littlechild and Thompson [10], address the problem of distributing runway construction costs among players whose aircrafts require different runway lengths. Costs are encoded in a vector $\mathbf{c} = [c_1, \ldots, c_n]^\top$, where c_i corresponds to player i. The characteristic function is

$$v(S) = \max_{i \in S} c_i. \tag{5}$$

Classified as line-graph maintenance problems, these games have closed-form Shapley values [20], making them ideal benchmark instances for the numerical studies in Section 6.

2.5. Weighted Voting Games

Weighted voting games [6,16,21] represent voting bodies in which players possess different voting weights (e.g., parliamentary seats), collected in $\mathbf{c} = [c_1, \ldots, c_n]^\top$. A coalition S succeeds when $\sum_{i \in S} c_i \geq C$ for a predetermined quota C. This yields a characteristic function defined by

$$v(S) = \begin{cases} 0 & \text{if } \sum_{i \in S} c_i < C \\ 1 & \text{if } \sum_{i \in S} c_i \geq C. \end{cases} \tag{6}$$

In our theoretical and numerical investigations in Sections 5 and 6, we frequently use a family of weighted voting games with $n = 7$ players and

$$\mathbf{c} = [1, 2, 3, 1, 1, 1, 1]^\top, \quad C \in \{1, \ldots, 10\}. \tag{7}$$

There exist fast methods for computing point-valued solutions of weighted voting games for large player numbers, for instance dynamic programming algorithms [22,23]. Hence these games serve as excellent test games for our experiments in Section 6.

2.6. Baseline Shapley (BShap) for Interpretable Machine Learning

Our numerical experiments in Section 6 apply the Shapley value framework to attribute a model's prediction to its input features. Here, the prediction task is interpreted as a cooperative game with features as players. Given a model o and an input x, the characteristic function $v(S)$ specifies the value of a feature subset S.

We adopt *Baseline Shapley (BShap)*, a method originally suggested in [1] and formalized by Sundararajan and Najmi [24]. The value $v(S)$ is defined deterministically relative to a *single baseline vector* z', where features not in S are set to their baseline counterparts:

$$v_{\text{BShap}}(S) = \mathsf{o}([\mathsf{z}_S, \mathsf{z}'_{\bar{S}}]) - \mathsf{o}(\mathsf{z}') \tag{8}$$

with $\bar{S}$ denoting the set of features not in S. The subtrahend $\mathsf{o}(\mathsf{z}')$ in (8) ensures the normalization $v_{\text{BShap}}(\emptyset) = 0$ as introduced in Subsection 2.1. Note that this normalization would not be needed for the algorithms based on marginal contributions investigated in our article [15], but becomes crucial for the algorithms based on least square formulations of the Shapley value discussed in Section 5.

Our specific baseline: In the applications in Section 6, the baseline z' is taken as the expected feature vector over the training distribution:

$$\mathsf{z}' = \bar{\mathsf{z}} = \mathbb{E}_{\mathsf{z} \sim \mathcal{D}}[\mathsf{z}].$$

with $\mathcal{D}$ representing the training data distribution. This anchors Shapley values to the prediction $\mathsf{o}(\bar{\mathsf{z}})$ of an average sample, providing a natural reference point. The resulting attributions thus explain the deviation of $\mathsf{o}(\mathsf{x})$ from this baseline.

 doi:10.20944/preprints202601.2308.v1

3. Preliminaries on Monte Carlo Methods for Estimation

In this section, we first introduce the use of Monte Carlo methods for estimating expectations together with two techniques for variance reduction. We draw mainly from the short overviews provided by Rubinstein and Kroese [25] and Botev and Ridder [26] as well as from the textbook by Rubinstein and Kroese [27].

Let $\mathbf{X}$ be a d-dimensional discrete (or continuous) random variable with a sample space $\mathcal{X} \subseteq \mathbb{R}^d$ and a probability mass function $q_{\mathbf{X}}$ (or probability density function, respectively), which may be known or unknown. Denote a specific realization of $\mathbf{X}$ by $\mathbf{x} \in \mathcal{X}$. For a function $H : \mathcal{X} \to \mathbb{R}$, we want to estimate the expectation

$$\mu = \mathbb{E}[H(\mathbf{X})]. \tag{9}$$

This expectation is frequently intractable in closed form, either because $q_{\mathbf{X}}$ is unknown (and only i.i.d. samples are given) or because the defining sum or integral is computationally infeasible. Monte Carlo methods circumvent this analytical hurdle through simulation.

We begin with Crude Monte Carlo as the foundational method, then present importance sampling and stratified sampling for variance reduction which we examine in this paper. A finite variance of $H(\mathbf{X})$ is assumed throughout this manuscript.

3.1. Crude Monte Carlo Method

Crude Monte Carlo proceeds by drawing an i.i.d. sample $\mathbf{X}_1, \ldots, \mathbf{X}_\tau \sim_{\text{iid}} q_{\mathbf{X}}$ and averaging the $H(\mathbf{X}_k)$ values. This yields the standard estimator [26]:

$$\hat{\mu} = \frac{1}{\tau} \sum_{k=1}^{\tau} H(\mathbf{X}_k). \tag{10}$$

According to [26], the estimator is unbiased, i.e. $\mathbb{E}[\hat{\mu}] = \mu$, and its variance is

$$\operatorname{Var}[\hat{\mu}] = \frac{1}{\tau} \operatorname{Var}[H(\mathbf{X})]. \tag{11}$$

3.2. Stratified Sampling for Variance Reduction

Stratified sampling is a well-established Monte Carlo variance reduction technique [26]. It separates the sample space $\mathcal{X}$ into ℓ disjoint strata $\{\mathcal{X}_1, \ldots, \mathcal{X}_\ell\}$ such that $\bigcup_{l=1}^{\ell} \mathcal{X}_l = \mathcal{X}$ with $\mathcal{X}_l \cap \mathcal{X}_{l'} = \emptyset$ for $l \neq l'$. Suppose L is a discrete random variable taking values in $\{1, \ldots, \ell\}$ with known probabilities $p_L(l) = \mathbb{P}(L = l) = \mathbb{P}(\mathbf{X} \in \mathcal{X}_l)$. We can rewrite μ as

$$\mu = \mathbb{E}_{p_L}\left[\mathbb{E}_{q_{\mathbf{X}}}[H(\mathbf{X}) \mid \mathbf{X} \in \mathcal{X}_L]\right] = \sum_{l=1}^{\ell} p_L(l)\, \mathbb{E}[H(\mathbf{X}) \mid \mathbf{X} \in \mathcal{X}_l] = \sum_{l=1}^{\ell} p_L(l)\, \mu_l,$$

with $\mu_l = \mathbb{E}[H(\mathbf{X}) \mid \mathbf{X} \in \mathcal{X}_l]$ standing for the expectation over the conditional probability distribution of $\mathbf{X}$ given that $\mathbf{X} \in \mathcal{X}_l$. Our stratified estimator of μ becomes

$$\hat{\mu}^{st} = \sum_{l=1}^{\ell} p_L(l)\, \hat{\mu}_l,$$

where $\hat{\mu}_l$ denotes the estimated value of $H(\mathbf{X})$ in stratum $\mathcal{X}_l$, i.e.,

$$\hat{\mu}_l = \frac{1}{\tau_l} \sum_{k=1}^{\tau_l} H(\mathbf{X}_{l,k}), \tag{12}$$

with the sample $\mathbf{X}_{l,1}, \ldots, \mathbf{X}_{l,\tau_l}$ of size τ_l being drawn i.i.d. from the conditional probability distribution of $\mathbf{X}$ given that $\mathbf{X} \in \mathcal{X}_l$. To ensure fair comparisons to other Monte Carlo techniques, we always assume

 doi:10.20944/preprints202601.2308.v1

$\sum_{l=1}^{\ell} \tau_l \approx \tau$ (up to rounding errors), where τ stands for the overall sample budget. The estimator is unbiased [26], i.e.,

$$\mathbb{E}[\hat{\mu}^{st}] = \mu.$$

Assuming that the sample sizes per stratum are proportionally assigned, i.e., $\tau_l = p_L(l)\,\tau$, the variance of the crude Monte Carlo estimator (11) has been established as an upper bound [26], i.e.

$$\mathrm{Var}[\hat{\mu}^{st}] \leq \mathrm{Var}[\hat{\mu}],$$

meaning that the stratified estimator $\hat{\mu}^{st}$ never performs worse than its crude counterpart $\hat{\mu}$.

3.3. Importance Sampling for Variance Reduction

Importance sampling introduces another probability mass (or density) function $p_{\mathbf{X}}$, such that $p_{\mathbf{X}}(\mathbf{x}) = 0 \implies H(\mathbf{x})\, q_{\mathbf{X}}(\mathbf{x}) = 0$. This allows (9) to be rewritten as

$$\mu = \mathbb{E}_{p_{\mathbf{X}}}\left[\frac{q_{\mathbf{X}}(\mathbf{X})}{p_{\mathbf{X}}(\mathbf{X})}\, H(\mathbf{X})\right]. \tag{13}$$

A Monte Carlo approximation based on (13) is

$$\hat{\mu} = \frac{1}{\tau}\sum_{k=1}^{\tau} \frac{q_{\mathbf{X}}(\mathbf{X}_k)}{p_{\mathbf{X}}(\mathbf{X}_k)}\, H(\mathbf{X}_k), \tag{14}$$

where $\mathbf{X}_1, \ldots, \mathbf{X}_\tau \sim_{\text{iid}} p_{\mathbf{X}}$. As shown in [25], this estimator is unbiased, i.e. $\mathbb{E}_{p_{\mathbf{X}}}[\hat{\mu}] = \mu$, and its variance is analogous to the crude Monte Carlo case

$$\mathrm{Var}_{p_{\mathbf{X}}}[\hat{\mu}] = \frac{1}{\tau}\,\mathrm{Var}_{p_{\mathbf{X}}}\left[\frac{q_{\mathbf{X}}(\mathbf{X})}{p_{\mathbf{X}}(\mathbf{X})}\, H(\mathbf{X})\right]. \tag{15}$$

With a properly selected $p_{\mathbf{X}}$, the variance of the importance sampling estimator is less than or equal to that of the crude Monte Carlo estimator. This follows from comparing (11) and (15), yielding

$$\frac{1}{\tau}\,\mathrm{Var}_{q_{\mathbf{X}}}[H(\mathbf{X})] \leq \min_{p_{\mathbf{X}}} \frac{1}{\tau}\,\mathrm{Var}_{p_{\mathbf{X}}}\left[\frac{q_{\mathbf{X}}(\mathbf{X})}{p_{\mathbf{X}}(\mathbf{X})}\, H(\mathbf{X})\right],$$

which holds because $p_{\mathbf{X}} = q_{\mathbf{X}}$ can always be chosen as the trivial case. We apply importance sampling in the context of TU games in Section 4.

4. Approximating Linear Solution Concepts via Importance Sampling

The exact calculation of a linear solution concept, i.e., (1), for a TU game normally requires summing up terms whose number grows exponentially in the number of players n. Therefore, approximation algorithms are needed to estimate these values in real-world situations, especially in the context of applications in interpretable machine learning [1,17–19] where one can typically not exploit any special structure of the underlying game for computing Shapley values exactly. In this section, we briefly summarize some ideas from [14] and the general framework for importance sampling on the coalition space for linear solution concepts which we recently proposed in [15] and will apply in Section 5.

Benati et al. [14] consider a uniform sampling strategy on the coalition space for approximating linear solution concepts, i.e. they simply adopt the uniform distribution $q = \mathcal{U}(2^N)$. Let us regard it as the crude Monte Carlo method on the coalition space. Although sampling subsets from the uniform distribution is both straightforward and unbiased, it is obviously not an optimal — or even recommendable — sampling strategy for all linear solution concepts or problem settings. For example, when approximating the Shapley value by sampling coalitions using (3), small or large coalitions obtain larger weights resulting in a strong influence on the estimator. However, when sampling

 doi:10.20944/preprints202601.2308.v1

uniformly from the coalition space, they may extract only a few samples resulting in a less accurate estimator.

To enable more efficient sampling, we apply the importance sampling technique from Subsection 3.3. By appropriately reweighting the estimator, importance sampling allows us to draw samples from a non-uniform, user-defined distribution while maintaining an unbiased estimate of the underlying linear solution concept.

Theorem 1. *[15] For all $i \in N$, let*

$$p_i : 2^N \to [0,1] \quad \textit{with} \quad p_i(S) = 0 \implies w_i(S)\,\nu_i(S) = 0$$

be a probability distribution and $\mathbb{S}_i \sim_{\text{iid}} p_i$ with $\mathbb{S}_i = [\mathcal{S}_{i,1}, \ldots, \mathcal{S}_{i,\tau}]^\top$ be a sample of size τ generated by sampling with replacement according to p_i. Then,

$$\hat{\boldsymbol{\alpha}}(N,v) = [\hat{\alpha}_i(N,v)]_{i\in N} = \left[\frac{1}{\tau}\sum_{\mathcal{S}\in\mathbb{S}_i}\frac{w_i(\mathcal{S})}{p_i(\mathcal{S})}\,\nu_i(\mathcal{S},v)\right]_{i\in N} \tag{16}$$

is an importance sampling estimator of the linear solution concept $\boldsymbol{\alpha}(N,v)$.

The following proposition connects our importance sampling estimator from Theorem 1 established in Pollmann and Staudacher [15] to a finding from Benati et al. [14], p. 95.

Proposition 1. *The importance sampling estimator from Theorem 1 has the following properties:*

(a) *It is unbiased, i.e.,*

$$\mathbb{E}[\hat{\alpha}_i(N,v)] = \alpha_i(N,v) \quad (\forall i \in N). \tag{17}$$

(b) *Its variance is given by*

$$\operatorname{Var}[\hat{\alpha}_i(N,v)] = \frac{1}{\tau}\left(\sum_{S\subseteq N}\frac{w_i(S)^2}{p_i(S)}\,\nu_i(S,v)^2 - \alpha_i(N,v)^2\right) \quad (\forall i \in N). \tag{18}$$

(c) *It is consistent in probability, i.e.,*

$$\hat{\alpha}_i(N,v) \xrightarrow{\tau\to\infty} \alpha_i(N,v) \quad (\forall i \in N). \tag{19}$$

We note that Theorem 1 and Proposition 1 subsume both the crude Monte Carlo method (setting $p_i(S) = q(S) = 2^{-n}$) and stratified sampling. The latter is covered because any stratum estimator for a linear solution concept can be written in the form of (16), allowing direct application of these results.

5. Least Squares Approaches for Approximating Shapley Values

This section covers the idea of approaching the approximation of Shapley values as a least squares problem. First, we give an introduction to the family of least square values in Section 5.1, followed by an overview about Shapley value approximation algorithms in this setting in Section 5.2. In Sections 5.3, 5.4, and 5.5, we compare these algorithms and put them into the overall context.

5.1. The Family of least Square Values

The *family of least square values* was proposed by Ruiz et al. [9] and comprises a generalization of the formulation of the Shapley value as a least squares problem, which was already proposed earlier by Charnes et al. [8].

Essentially, this family consists of all solution concepts that can be obtained as minimizers of a weighted least squares problem, where the weights determine the relative importance of different coalition sizes. These weights are given by the function $m : \{0, \ldots, n\} \to \mathbb{R}_{\geq 0}$ that assigns a weight

 doi:10.20944/preprints202601.2308.v1

to each coalition size $s \in \{0, \ldots, n\}$ with $m(s) > 0$ for at least one s. Clearly, per definition, m is symmetric in a sense of assigning the same weight to coalitions of the same cardinality in order to retrieve symmetric solutions like the Shapley value. We denote by $\mathbf{y} \in \mathbb{R}^n$ a payoff vector for a given game with y_i being the payoff for player i. In the following, we use the shorthand notation $\mathbf{y}(S) = \sum_{i \in S} y_i$. Recall that a payoff vector is called *efficient* if $\mathbf{y}(N) = v(N)$.

With these definitions established, any solution concept belonging to the family of least square values can be expressed as the optimizer of the following weighted least squares problem:

$$\begin{aligned} \min_{\mathbf{y} \in \mathbb{R}^n} \quad & \sum_{S \subseteq N} m(|S|) \left(v(S) - \mathbf{y}(S)\right)^2 \\ \text{s.t.} \quad & \mathbf{y}(N) = v(N), \end{aligned} \tag{20}$$

with m being the weight function that is specific to the particular solution concept.

The closed-form solution — again, for a fixed m — to problem (20) is

$$y_i = \frac{v(N)}{n} + \frac{1}{\gamma n} \left(n a_i - \sum_{j \in N} a_j \right) \quad (\forall i \in N), \tag{21}$$

with

$$a_j = \sum_{\substack{S \subseteq N \\ j \in S}} m(|S|)\, v(S) \quad \text{and} \quad \gamma = \sum_{s=1}^{n-1} m(s) \binom{n-2}{s-1}. \tag{22}$$

These expressions illustrate how the resulting allocation depends on the weight function m.

Least square values as linear solution concepts

Let us briefly point out why all members of the family of least square values are linear solution concepts in the sense of the definition from Subsection 2.2. Benati et al. [14] showed that (21) can be rewritten as

$$\mathbf{y} = \mathbf{1} \frac{v(N)}{n} + \sum_{\substack{S \subseteq N \\ 0 < |S| < n}} \boldsymbol{w}(S)\, v(S), \tag{23}$$

where $\mathbf{1} \in \mathbb{R}^n$ is the all-ones vector and the individual elements of $\boldsymbol{w}(S)$ are given by

$$w_i(S) = \begin{cases} \frac{(n-|S|)\, m(|S|)}{n\gamma} & \text{if } i \in S \\ -\frac{|S|\, m(|S|)}{n\gamma} & \text{if } i \notin S. \end{cases} \tag{24}$$

From (23), it follows directly that this representation satisfies the definition of linear solution concepts provided in Subsection 2.2.

The Shapley value in the context of least square values

The family of least square values was introduced by Ruiz et al. [9] after Charnes et al. [8] had demonstrated that the Shapley value can be expressed as a least squares optimization problem. In detail, the vector of Shapley values $\boldsymbol{\phi}$ is the solution to problem (20) when m is defined as

$$m(s) = \frac{1}{n-1} \binom{n-2}{s-1}^{-1}, \tag{25}$$

and therefore, via (22), one obtains

$$\gamma = \sum_{s=1}^{n-1} m(s) \binom{n-2}{s-1} = \sum_{s=1}^{n-1} \frac{1}{n-1} \binom{n-2}{s-1}^{-1} \binom{n-2}{s-1} = 1. \tag{26}$$

 doi:10.20944/preprints202601.2308.v1

Thus, by inserting (26) into (21) and reformulating the result, one obtains the closed-form solution for the Shapley value as

$$\phi_i = \frac{v(N)}{n} + \frac{n-1}{n} a_i - \frac{1}{n} \sum_{\substack{j \in N \\ j \neq i}} a_j \quad (\forall i \in N), \tag{27}$$

with a_j given by (22) for all $j \in N$. We illustrate the computation of the Shapley value as a least squares optimization problem in Figure 1.

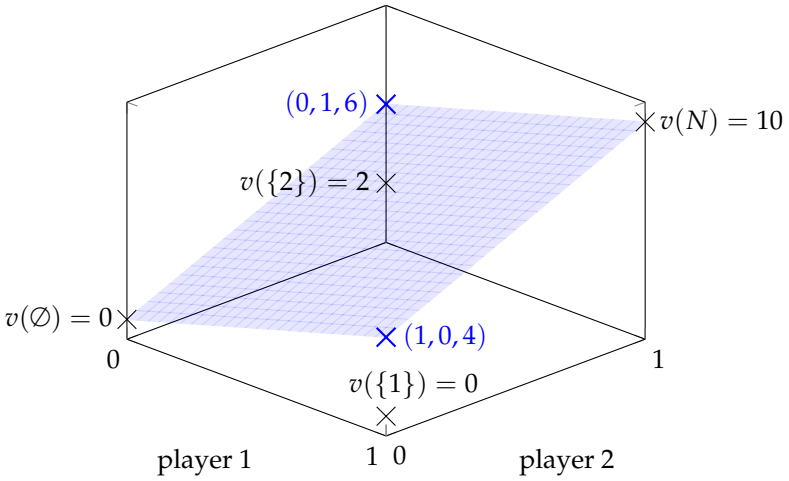

Figure 1. Let (N, v) be a cooperative game with $N = \{1, 2\}$ and $v(\emptyset) = 0$, $v(\{1\}) = 0$, $v(\{2\}) = 2$, and $v(\{1, 2\}) = 10$. The *player 1* and *player 2* axes should be interpreted in a discrete way such that 0 denotes the exclusion and 1 specifies the inclusion of a player. The blue plane is the solution to problem (20) with m being defined as (25) and $\gamma = 1$ being obtained from (26). As a result, the coefficients defining this plane are the Shapley values of the game (N, v), i.e., $\phi_1 = 4$, and $\phi_2 = 6$.

5.2. Shapley Value Approximation Algorithms

In this subsection, we provide an introduction to Shapley value approximation algorithms that are based on the formulation of the Shapley value as a least squares problem. The algorithms discussed below are not meant to be an exhaustive list, but rather represent those algorithms we will analyse and test in the rest of the article. Specifically, the first three algorithms have been proposed by Benati et al. [14] in the context of stochastic approximations of cooperative games and are based on the family of least square values defined in Subsection 5.1, while the latter two have their origins in the machine learning world.

Additionally, we would like to draw attention to the absence of the projection method proposed by Benati et al. [14], which ensures efficiency and symmetry of the obtained estimators. Although this approach potentially improves the accuracy of the estimators (or at least never makes their accuracy worse), it is not agnostic to the underlying characteristic function, as it requires prior knowledge about symmetric players, which is an information that we assume to be unavailable — in particular, in the context of interpretable machine learning. Nevertheless, one could still achieve estimator efficiency through a single evaluation of $v(N)$, similar to Castro et al. [28], who proposed filling the "efficiency gap" post hoc. However, since such additional considerations lie outside the scope of this work, we exclude them equally from all algorithms.

Least Squares Sampling (LSS)

This paragraph deals with an algorithm proposed by Benati et al. [14] that approximates (23) by using importance sampling as defined in Theorem 1. Specifically, we refer to the estimator in equation (22) on p. 97 in the paper [14] combined with their weighted sampling strategy. In the following, we name this algorithm *LSS*.

 doi:10.20944/preprints202601.2308.v1

The idea of LSS is to sample coalitions with replacement according to

$$p(S) = \begin{cases} \frac{1}{(n-1)\binom{n}{|S|}} & \text{if } 0 < |S| < n \\ 0 & \text{else,} \end{cases} \tag{28}$$

such that, by combining (24), (25), (26), and (28), one obtains

$$\frac{w_i(S)}{p(S)} = \begin{cases} \frac{n-1}{|S|} & \text{if } i \in S \\ -\frac{n-1}{n-|S|} & \text{if } i \notin S. \end{cases} \tag{29}$$

Then, based on Theorem 1 as well as (23) and (29), an approximation of the Shapley value of player i is given by

$$\hat{\phi}_i = \frac{v(N)}{n} + \frac{1}{\tau} \sum_{\mathcal{S} \in \mathbb{S}} \frac{w_i(\mathcal{S})}{p(\mathcal{S})} v(\mathcal{S}) \tag{30}$$

$$= \frac{v(N)}{n} + \frac{1}{\tau} \left(\sum_{\substack{\mathcal{S} \in \mathbb{S} \\ i \in \mathcal{S}}} \frac{n-1}{|\mathcal{S}|} v(\mathcal{S}) - \sum_{\substack{\mathcal{S} \in \mathbb{S} \\ i \notin \mathcal{S}}} \frac{n-1}{n-|\mathcal{S}|} v(\mathcal{S}) \right), \tag{31}$$

where $\mathbb{S}$ is obtained by sampling τ times with replacement according to p.

Equation (31) serves as the basis for Algorithm 1. Note that one evaluation of v can be used to update all players, reducing the number of evaluations of v and increasing convergence speed.

Algorithm 1 LSS

$\hat{\boldsymbol{\phi}} \leftarrow \mathbf{1} \frac{v(N)}{n}$
for τ **times do**
  Draw $\mathcal{S} \sim p$
  $v_{\mathcal{S}} \leftarrow v(\mathcal{S})$
  $w_{\text{in}} \leftarrow \frac{n-1}{|\mathcal{S}|}$
  $w_{\text{out}} \leftarrow \frac{n-1}{n-|\mathcal{S}|}$
  for all $i \in N$ **do**
    if $i \in \mathcal{S}$ **then**
      $\hat{\phi}_i \leftarrow \hat{\phi}_i + \frac{1}{\tau} w_{\text{in}} v_{\mathcal{S}}$
    else
      $\hat{\phi}_i \leftarrow \hat{\phi}_i - \frac{1}{\tau} w_{\text{out}} v_{\mathcal{S}}$
    end if
  end for
end for

Since (31) is an importance sampling estimator of a linear solution concept in the sense of Theorem 1, which can easily be obtained via (23) and (30), LSS inherits the properties stated in Proposition 1. Thus, the estimator is unbiased, and we can use (18) to calculate its variance:

Proposition 2. *The variance of the LSS estimator is given by*

$$\text{Var}[\hat{\phi}_i] = \frac{1}{\tau} \left(\sum_{\substack{S \subseteq N \\ 0<|S|<n \\ i \in S}} \frac{n-|S|}{n\,|S|\binom{n-2}{|S|-1}} v(S)^2 + \sum_{\substack{S \subseteq N \\ 0<|S|<n \\ i \notin S}} \frac{|S|}{n\,(n-|S|)\binom{n-2}{|S|-1}} v(S)^2 - \alpha_i^2 \right) \tag{32}$$

with

$$\alpha_i = \phi_i - \frac{v(N)}{n}$$

for all $i \in N$.

 doi:10.20944/preprints202601.2308.v1

Proof. Using (18) and setting $\nu_i = v$ results in

$$\mathrm{Var}[\hat{\phi}_i] = \frac{1}{\tau}\left(\sum_{\substack{S \subseteq N \\ 0<|S|<n}} \frac{w_i(S)^2}{p(S)}\, v(S)^2 - \alpha_i^2 \right), \tag{33}$$

where $\alpha_i = \phi_i - \frac{v(N)}{n}$. Here, the term $\frac{v(N)}{n}$ is excluded from the variance calculation since it is added once at the beginning of the LSS algorithm, but it is not part of the sampling procedure and thus, there is no randomness to it, and it does not influence the variance of $\hat{\phi}_i$. As a result, we construct a temporary solution concept of the game, α_i, that does not contain the constant term $\frac{v(N)}{n}$.

With that given, one obtains (32) by calculating

$$\frac{w_i(S)^2}{p(S)} = \begin{cases} \frac{n-|S|}{n\,|S|}\binom{n-2}{|S|-1}^{-1} & \text{if } i \in S \\ \frac{|S|}{n\,(n-|S|)}\binom{n-2}{|S|-1}^{-1} & \text{if } i \notin S, \end{cases}$$

and inserting the result into (33). □

Stratified Least Squares Sampling (S-LSS)

This paragraph discusses another algorithm proposed by Benati et al. [14], which approximates the Shapley value through what the authors introduce as their stratification strategy. Specifically, we refer to equations (23) and (24) on p. 97 in the paper [14] without any reuse of samples across strata. In the following, we refer to this method as *S-LSS*.

The stratification according to [14] comes into play when approximating the individual addends a_j in (21), where a reformulation of (22) results in

$$a_j = \sum_{s=1}^{n-1} \underbrace{\sum_{\substack{S \subseteq N \\ s=|S| \\ j \in S}} m(s)\, v(S)}_{a_{j,s}}. \tag{34}$$

Based on that formulation, approximations of $a_{j,s}$, denoted by $\hat{a}_{j,s}$, are obtained as

$$\hat{a}_{j,s} = \frac{1}{\tau_{j,s}} \sum_{\mathcal{S} \in \mathbb{S}_{j,s}} \frac{m(s)}{p_{j,s}(\mathcal{S})}\, v(\mathcal{S}) = \frac{1}{\tau_{j,s}\,(n-s)} \sum_{\mathcal{S} \in \mathbb{S}_{j,s}} v(\mathcal{S}), \tag{35}$$

with $\mathbb{S}_{j,s}$ being an i.i.d. sample of size $\tau_{j,s}$ from $\{S \subseteq N \mid (s = |S|) \wedge (j \in S)\}$ according to the uniform probability distribution $p_{j,s}(S) = \binom{n-1}{s-1}^{-1}$.

Then, based on (27), one obtains an estimator of ϕ_i as

$$\hat{\phi}_i = \frac{v(N)}{n} + \frac{n-1}{n} \sum_{s=1}^{n-1} \hat{a}_{i,s} - \frac{1}{n} \sum_{\substack{j \in N \\ j \neq i}} \sum_{s=1}^{n-1} \hat{a}_{j,s},$$

forming the basis for Algorithm 2.

 doi:10.20944/preprints202601.2308.v1

Algorithm 2 S-LSS

for $i \in N$ **do**
  for $s = 1$ **to** $n-1$ **do**
    $\hat{a}_{i,s} \leftarrow 0$
    for $\tau_{i,s}$ **times do**
      Draw $\mathcal{S} \sim p_{i,s}$
      $\hat{a}_{i,s} \leftarrow \hat{a}_{i,s} + \frac{1}{\tau_{i,s}\,(n-s)}\, v(\mathcal{S})$
    end for
  end for
end for
$\hat{\phi}_i = \frac{v(N)}{n} + \frac{n-1}{n} \sum_{s=1}^{n-1} \hat{a}_{i,s} - \frac{1}{n} \sum_{j \in N\,:\, j \neq i} \sum_{s=1}^{n-1} \hat{a}_{j,s} \quad (\forall i \in N)$

Clearly, by using the formulation given in (35), the estimators $\hat{a}_{j,s}$ can be seen as estimators of individual solution concepts $a_{j,s}$ in the sense of Theorem 1 by setting

$$w_j(S) = \begin{cases} m(s) & \text{if } (s = |S|) \wedge (j \in S) \\ 0 & \text{else,} \end{cases}$$

as well as

$$p_j(S) = \begin{cases} p_{j,s}(S) & \text{if } (s = |S|) \wedge (j \in S) \\ 0 & \text{else,} \end{cases}$$

and $\nu_j = v$.

Therefore, the individual stratum estimators are unbiased, and (18) can be used to calculate their variances:

Proposition 3. *In the context of the S-LSS algorithm, the variance of $\hat{a}_{j,s}$ is given by*

$$\mathrm{Var}[\hat{a}_{j,s}] = \frac{1}{\tau_{j,s}} \left(\frac{1}{(n-s)(n-1)\binom{n-2}{s-1}} \sum_{\substack{S \subseteq N \\ s=|S| \\ j \in S}} v(S)^2 - a_{j,s}^2 \right) \tag{36}$$

for all $j \in N$ and $s \in \{1, \ldots, n-1\}$.

Proof. By using the fact that

$$\frac{\binom{n-1}{s-1}}{\binom{n-2}{s-1}} = \frac{(n-1)!}{(s-1)!\,(n-s)!} \frac{(s-1)!\,(n-s-1)!}{(n-2)!} = \frac{n-1}{n-s}, \tag{37}$$

we can use (18) to obtain

$$\begin{aligned} \sum_{\substack{S \subseteq N \\ s=|S| \\ j \in S}} \frac{m(s)^2}{p_{j,s}(S)}\, v(S)^2 &= \frac{\binom{n-1}{s-1}}{(n-1)^2 \binom{n-2}{s-1}^2} \sum_{\substack{S \subseteq N \\ s=|S| \\ j \in S}} v(S)^2 \\ &= \frac{n-1}{(n-s)(n-1)^2\binom{n-2}{s-1}} \sum_{\substack{S \subseteq N \\ s=|S| \\ j \in S}} v(S)^2 \\ &= \frac{1}{(n-s)(n-1)\binom{n-2}{s-1}} \sum_{\substack{S \subseteq N \\ s=|S| \\ j \in S}} v(S)^2. \end{aligned}$$

 doi:10.20944/preprints202601.2308.v1

Inserting this result back into (18) results in (36). □

Based on the individual stratum estimator variances given by Proposition 3, Benati et al. [14] stated the overall variance of the estimated Shapley value of player i as

$$\begin{aligned}\mathrm{Var}[\hat{\phi}_i] &= \mathrm{Var}\left[\frac{v(N)}{n} + \frac{n-1}{n}\sum_{s=1}^{n-1}\hat{a}_{i,s} - \frac{1}{n}\sum_{\substack{j\in N\\ j\neq i}}\sum_{s=1}^{n-1}\hat{a}_{j,s}\right] \\ &= \left(\frac{n-1}{n}\right)^2 \sum_{s=1}^{n-1}\mathrm{Var}[\hat{a}_{i,s}] + \frac{1}{n^2}\sum_{\substack{j\in N\\ j\neq i}}\sum_{s=1}^{n-1}\mathrm{Var}[\hat{a}_{j,s}]. \end{aligned} \tag{38}$$

Perhaps surprisingly and contrary to what might be expected from the definition of stratified sampling in Subsection 3.2, S-LSS does not guarantee a smaller variance compared to LSS, as we discuss in Subsection 5.3 and show empirically in Section 6.

Sample Reuse Stratified Least Squares Sampling (SRS-LSS)

Benati et al. [14] proposed a third algorithm, which is based on the S-LSS algorithm and reuses samples across strata. Specifically, we refer to the final three sentences in subsection 4.2 on p. 97 in the paper [14]. In the following, we refer to this method as *SRS-LSS*.

The central idea is to reuse one evaluation $v(\mathcal{S})$ in (35) to update all stratum estimators $\hat{a}_{j,s}$ where $s = |\mathcal{S}|$ and $j \in \mathcal{S}$, hence the name. To achieve that, samples $\mathbb{S}_s$ of size τ_s are drawn from $\{S \subseteq N \mid s = |S|\}$ with probabilities $p_s(S) = \binom{n}{s}^{-1}$. Based on the elements of $\mathcal{S} \in \mathbb{S}_s$, all stratum estimators $\hat{a}_{j,s}$ with $j \in \mathcal{S}$ are updated with a single evaluation $v(\mathcal{S})$. As a side effect, the number of evaluations of v that are used to update player j's stratum, i.e., $\tau_{j,s}$, are dependent on the generated samples, which is why, in Algorithm 3, we need to keep track of their values as well. This essentially means that all $\tau_{j,s}$ become estimators themselves, which is why we denote them by $\hat{\tau}_{j,s}$. Additionally, we use $\tilde{a}_{j,s}$ to denote unnormalized estimators of $a_{j,s}$ before dividing by $\hat{\tau}_{j,s}\,(n-s)$.

Algorithm 3 SRS-LSS

for $s = 1$ **to** $n-1$ **do**
  $\hat{\tau}_{i,s} \leftarrow 0 \quad (\forall i \in N)$
  $\tilde{a}_{i,s} \leftarrow 0 \quad (\forall i \in N)$
  for τ_s **times do**
    Draw $\mathcal{S} \sim p_s$
    $v_{\mathcal{S}} \leftarrow v(\mathcal{S})$
    for all $i \in \mathcal{S}$ **do**
      $\tilde{a}_{i,s} \leftarrow \tilde{a}_{i,s} + v_{\mathcal{S}}$
      $\hat{\tau}_{i,s} \leftarrow \hat{\tau}_{i,s} + 1$
    end for
  end for
  $\hat{a}_{i,s} \leftarrow \frac{1}{\hat{\tau}_{i,s}\,(n-s)}\,\tilde{a}_{i,s} \quad (\forall i \in N)$
end for
$\hat{\phi}_i = \frac{v(N)}{n} + \frac{n-1}{n}\sum_{s=1}^{n-1}\hat{a}_{i,s} - \frac{1}{n}\sum_{j\in N: j\neq i}\sum_{s=1}^{n-1}\hat{a}_{j,s} \quad (\forall i \in N)$

One needs to guarantee that $\hat{\tau}_{j,s} > 0$ for all $j \in N$ and $s \in \{1, \ldots, n-1\}$ in order for SRS-LSS to function properly and avoid divisions by zero. One straightforward approach is to repeatedly generate samples until this condition is satisfied, or alternatively, to employ a conditional probability distribution that accounts for which elements have not yet been sampled. In the remainder of this work, we rely on the results from the following proposition, and therefore, we do not incorporate additional safeguards into Algorithm 3.

 doi:10.20944/preprints202601.2308.v1

Proposition 4. *Let $\tau_s \to \infty$ as $\tau \to \infty$ for all $s \in \{1, \dots, n-1\}$. Then, with probability approaching one asymptotically, each stratum estimator receives at least one sample, i.e.,*

$$\lim_{\tau \to \infty} \mathbb{P}(\exists j \in N, s \in \{1, \dots, n-1\} : \hat{\tau}_{j,s} = 0) = 0.$$

Proof. The probability of a player $j \in N$ belonging to a random coalition of size $s \in \{1, \dots, n-1\}$ is given by $\frac{s}{n} > 0$. Thus, we have

$$\mathbb{P}(\hat{\tau}_{j,s} = 0) = \left(1 - \frac{s}{n}\right)^{\tau_s} \xrightarrow{\tau_s \to \infty} 0. \tag{39}$$

Since

$$\mathbb{P}(\exists j \in N, s \in \{1, \dots, n-1\} : \hat{\tau}_{j,s} = 0) = \mathbb{P}\left(\bigcup_{j \in N} \bigcup_{s=1}^{n-1} \{\hat{\tau}_{j,s} = 0\} \right),$$

we can use the union bound [29] and (39) to derive

$$\mathbb{P}\left(\bigcup_{j \in N} \bigcup_{s=1}^{n-1} \{\hat{\tau}_{j,s} = 0\} \right) \leq \sum_{j \in N} \sum_{s=1}^{n-1} \mathbb{P}(\hat{\tau}_{j,s} = 0) \xrightarrow{\tau \to \infty} 0,$$

which holds as long as $\tau_s \to \infty$ with $\tau \to \infty$. □

Since Benati et al. [14] did not include a variance analysis of the SRS-LSS estimator in their work, we refer to Subsection 5.4 for a detailed discussion of the variance of the obtained estimator.

KernelSHAP (KS)

Let us now introduce the *KernelSHAP* algorithm, one of the predominant Shapley approximation algorithms in the machine learning world. It was developed by Lundberg and Lee [1], without knowledge of prior works like [8] and [9]. The optimization objective is

$$\begin{aligned} \boldsymbol{\phi} = \underset{\mathbf{y} \in \mathbb{R}^n}{\arg\min} \quad & \sum_{\substack{\mathbf{z} \in \{0,1\}^n \\ 0 < |\mathbf{z}| < n}} p(\mathbf{z}) \left(\mathbf{y}(\mathbf{z}) - v(\mathbf{z}) \right)^2 \\ \text{s.t.} \quad & \mathbf{y}(N) = v(N) \end{aligned} \tag{40}$$

to calculate the Shapley values for the cooperative game (N, v) [7]. Here, $p \propto \omega$ denotes a probability mass function satisfying $p(\mathbf{0}) = 0$ and $p(\mathbf{1}) = 0$ such that

$$p(\mathbf{z}) = \begin{cases} Q^{-1}\, \omega(\mathbf{z}) & \text{if } 0 < |\mathbf{z}| < n \\ 0 & \text{else,} \end{cases} \tag{41}$$

where

$$\omega(\mathbf{z}) = \frac{n-1}{\binom{n}{|\mathbf{z}|} |\mathbf{z}| (n - |\mathbf{z}|)} \tag{42}$$

is the Shapley kernel and

$$Q = (n-1) \sum_{s=1}^{n-1} \frac{1}{s(n-s)} \tag{43}$$

is a normalization constant [7]. As noted in Subsection 2.1, all functions defined on $\{0,1\}^n$ are implicitly also defined on N.

 doi:10.20944/preprints202601.2308.v1

Since solving (40) requires optimizing over the elements in $\{0,1\}^n$, a set that grows exponentially in n, KernelSHAP approximates Shapley values by generating a sample $\mathbb{S} \sim_{\text{iid}} p$ of size τ and solving the approximate optimization problem defined over the sampled subsets only, i.e.,

$$\begin{aligned} \hat{\boldsymbol{\phi}} = \underset{\mathbf{y}\in\mathbb{R}^n}{\arg\min} \quad & \frac{1}{\tau}\sum_{\mathcal{S}\in\mathbb{S}} \big(\mathbf{y}(\mathcal{S}) - v(\mathcal{S})\big)^2 \\ \text{s.t.} \quad & \mathbf{y}(N) = v(N). \end{aligned} \tag{44}$$

Following [7], the Lagrangian of the approximate problem given by (44) with multiplier $\lambda \in \mathbb{R}$ is

$$\begin{aligned} \hat{\mathcal{L}}(\mathbf{y},\lambda) = \mathbf{y}^\top \underbrace{\left(\frac{1}{\tau}\sum_{\mathcal{S}\in\mathbb{S}} \mathbf{z}(\mathcal{S})\,\mathbf{z}(\mathcal{S})^\top\right)}_{\hat{A}} \mathbf{y} - 2\mathbf{y}^\top \underbrace{\left(\frac{1}{\tau}\sum_{\mathcal{S}\in\mathbb{S}} \mathbf{z}(\mathcal{S})\,v(\mathcal{S})\right)}_{\hat{\boldsymbol{b}}} \\ + \frac{1}{\tau}\sum_{\mathcal{S}\in\mathbb{S}} v(\mathcal{S})^2 + 2\lambda\,\big(\mathbf{y}(N) - v(N)\big). \end{aligned} \tag{45}$$

Based on the Lagrangian, Covert and Lee [7] derived the closed-form solution

$$\hat{\boldsymbol{\phi}} = \hat{A}^{-1}\left(\hat{\boldsymbol{b}} - \mathbf{1}\frac{\mathbf{1}^\top\hat{A}^{-1}\hat{\boldsymbol{b}} - v(N)}{\mathbf{1}^\top\hat{A}^{-1}\mathbf{1}}\right).$$

The resulting estimator is consistent in probability, but assessing its unbiasedness or variance is challenging due to terms like $\hat{A}^{-1}\hat{\boldsymbol{b}}$. While the unbiasedness or variance of $\hat{A}$ or $\hat{\boldsymbol{b}}$ can be analyzed individually [7], their interaction complicates the analysis of the final Shapley value estimator.

The KernelSHAP algorithm will not be further analyzed theoretically in this work. Due to its setup, it is unclear how the obtained estimator fits the framework defined by Theorem 1 or Monte Carlo estimators introduced in Section 3 in general. To the best of our knowledge, no analytic solution for its variance or a proof of its unbiasedness has been derived up today, which is why the *unbiased KernelSHAP* algorithm was introduced, guaranteeing unbiasedness and facilitating the variance analysis of the obtained Shapley value estimator.

Unbiased KernelSHAP (UKS)

The *UKS* algorithm [7] is a continued development of the KernelSHAP algorithm from the previous paragraph, where we pointed out that it is unclear if the KernelSHAP estimator is unbiased due to the interactions between $\hat{A}$ and $\hat{\boldsymbol{b}}$.

Considering the original optimization problem defined in (40), its Lagrangian with multiplier $\lambda \in \mathbb{R}$ is given by

$$\mathcal{L}(\mathbf{y},\lambda) = \mathbf{y}^\top \underbrace{\mathbb{E}[ZZ^\top]}_{A}\mathbf{y} - 2\mathbf{y}^\top \underbrace{\mathbb{E}[Zv(Z)]}_{\boldsymbol{b}} + \mathbb{E}[v(Z)^2] + 2\lambda\,\big(\mathbf{1}^\top\mathbf{y} - v(N)\big),$$

whereby Z is a random variable taking values from $\{0,1\}^n$ and being distributed according to p defined by (41). With that formulation given, the closed-form solution of the original problem can be derived as

$$\boldsymbol{\phi} = A^{-1}\left(\boldsymbol{b} - \mathbf{1}\frac{\mathbf{1}^\top A^{-1}\boldsymbol{b} - v(N)}{\mathbf{1}^\top A^{-1}\mathbf{1}}\right).$$

Clearly, $\boldsymbol{b}$ cannot be computed exactly for large n, since it depends on v, and thus, needs $2^n - 2$ evaluations of v in order to be exact. Nevertheless, since A does not depend on v and the probability distribution p is known, A can be pre-computed and is exact, which is the main improvement introduced by UKS in contrast to the original KernelSHAP algorithm.

 doi:10.20944/preprints202601.2308.v1

Thus, UKS estimates all players' Shapley values via

$$\hat{\boldsymbol{\phi}} = A^{-1}\left(\hat{\boldsymbol{b}} - \mathbf{1}\frac{\mathbf{1}^\top A^{-1}\hat{\boldsymbol{b}} - v(N)}{\mathbf{1}^\top A^{-1}\mathbf{1}}\right), \tag{46}$$

where $\hat{\boldsymbol{b}}$ is defined in (45) as for the KernelSHAP algorithm, i.e., $\hat{\boldsymbol{b}} = \frac{1}{\tau}\sum_{\mathcal{S}} \mathbf{z}(\mathcal{S})\, v(\mathcal{S})$, and $A \in \mathbb{R}^{n\times n}$ is given by

$$A = \begin{bmatrix} \frac{1}{2} & \rho & \cdots & \rho \\ \rho & \frac{1}{2} & \cdots & \rho \\ \vdots & \vdots & \ddots & \vdots \\ \rho & \rho & \cdots & \frac{1}{2} \end{bmatrix} \tag{47}$$

with

$$\rho = \frac{1}{n\,(n-1)} \frac{\sum_{s=2}^{n-1} \frac{s-1}{n-s}}{\sum_{s=1}^{n-1} \frac{1}{s\,(n-s)}}. \tag{48}$$

Following the results provided in [7], the resulting Shapley value estimator is unbiased and consistent in probability. Furthermore, its covariance is given by

$$\Sigma_{\hat{\boldsymbol{\phi}}} = E\Sigma_{\hat{\boldsymbol{b}}}E^\top \tag{49}$$

with

$$\Sigma_{\hat{\boldsymbol{b}}} = \mathrm{Cov}[Zv(Z)]$$

and

$$E = A^{-1} - \frac{A^{-1}\mathbf{1}\mathbf{1}^\top A^{-1}}{\mathbf{1}^\top A^{-1}\mathbf{1}}.$$

5.3. Comparison of LSS and S-LSS

In this subsection, we first clarify the relationship between the two algorithms LSS and S-LSS which we introduced at the beginning of Subsection 5.2, followed by a detailed comparison of variances.

Algorithm Comparison

For comparing the algorithms LSS and S-LSS, we start with the following proposition:

Proposition 5. *In the sense of the definition of stratified sampling provided in Section 3.2, S-LSS is not a valid stratification scheme of LSS.*

Proof. As stated in Section 3.2, stratification divides the sample space $\mathcal{X}$ into strata $\{\mathcal{X}_1, \ldots, \mathcal{X}_\ell\}$, such that $\mathcal{X}_l \cap \mathcal{X}_{l'} = \emptyset$ for $l, l' \in \{1, \ldots, \ell\}$ and $l \neq l'$.

By observing the strata definitions given in (34), one obtains that their sample spaces have the intersection

$$\{S \subseteq N \mid (s = |S|) \wedge (j \in S) \wedge (j' \in S)\} \quad (\forall j \neq j' \in N).$$

Clearly, for any coalition size $s \in \{2, \ldots, n-1\}$, this set is nonempty, since it contains at least one element of the form $\{j, j', \ldots\}$, whereby the dots indicate the presence of $s-2$ distinct players other than j and j'. Therefore, S-LSS does not form a valid partition of the sample space as required by our definition in Subsection 3.2. □

We note in passing that the structure of the overlap between strata is such that it does not introduce a bias in the Shapley estimator defined by S-LSS.

 doi:10.20944/preprints202601.2308.v1

In order to generate a deeper understanding regarding the relationship between LSS and S-LSS, we explain how Benati et al. [14] derived the LSS algorithm from the initial formulation given by (21). Via (21) and (22), one directly obtains

$$y_i = \frac{v(N)}{n} + \frac{1}{\gamma n}\left(n \underbrace{\sum_{\substack{S\subseteq N \\ i\in S}} m(|S|)\, v(S)}_{a_i} - \sum_{j\in N} \underbrace{\sum_{\substack{S\subseteq N \\ j\in S}} m(|S|)\, v(S)}_{a_j} \right) \tag{50}$$

$$= \frac{v(N)}{n} + \sum_{S\subseteq N} \frac{n\,\mathbb{1}_{i\in S}}{\gamma n}\, m(|S|)\, v(S) - \underbrace{\sum_{j\in N} \sum_{\substack{S\subseteq N \\ j\in S}} \frac{1}{\gamma n}\, m(|S|)\, v(S)}_{\sum_{S\subseteq N} \frac{|S|}{\gamma n}\, m(|S|)\, v(S)}$$

$$= \frac{v(N)}{n} + \sum_{S\subseteq N} \underbrace{\frac{n\,\mathbb{1}_{i\in S} - |S|}{\gamma n}\, m(|S|)}_{w_i(S)}\, v(S), \tag{51}$$

which matches the formulation proposed in (23) with weights given by (24). Therefore, one evaluation of v for a given S in (51) corresponds to updating all a_j where $j \in S$ in (50). This implicit update of all a_j where $j \in S$ is encapsulated by the weight function w_i.

As a result, stratifying by s and j, as done in the case of S-LSS, cannot be derived from LSS in the same way as defined in Section 3.2, since LSS is an algorithm that simultaneously updates *multiple* strata of S-LSS with a *single* sampled element, i.e., with *one* evaluation of v.

We conclude that the statement from Subsection 3.2, which points out that stratified sampling always yields a variance less than or equal to that of the crude Monte Carlo method, does not apply here, since S-LSS is not a valid stratification scheme of LSS. Therefore, in order to understand how their variances compare, we calculate both algorithms' variances on two selected examples.

Variance Comparison in the Absence of Variance Within Strata

First, let us compare the variances of the LSS and S-LSS estimator in the context of an example game characterized by large variance between strata and no variance within strata. In particular, we consider a cardinality game with $n \geq 2$ players and the characteristic function

$$v(S) = |S|. \tag{52}$$

Lemma 1. *On the game defined by (52) with $n \geq 2$, the variance of the LSS estimator is greater zero for all players, i.e.,*

$$\mathrm{Var}[\hat{\phi}_i^{\mathrm{LSS}}] > 0 \quad (\forall i \in N).$$

Proof. First, we obtain that the Shapley values of the game are given by $\frac{v(N)}{n}$ for all $i \in N$, since all i are symmetric. Without loss of generality, we fix one player $i \in N$ in order to simplify notations. Then, by using Proposition 2, we derive

$$\alpha_i^2 = \left(\phi_i - \frac{v(N)}{n}\right)^2 = 0.$$

Inserting α_i^2 into (32), multiplying by τ, and using the identities defined by (37) as well as

$$\frac{\binom{n-1}{s}}{\binom{n-2}{s-1}} = \frac{(n-1)!}{s!(n-s-1)!}\frac{(s-1)!(n-s-1)!}{(n-2)!} = \frac{n-1}{s}$$

 doi:10.20944/preprints202601.2308.v1

results in

$$\begin{aligned}\tau \operatorname{Var}[\hat{\phi}_i^{\mathrm{LSS}}] &= \sum_{\substack{S\subseteq N\\ 0<|S|<n\\ i\in S}} \frac{n-|S|}{n\,|S|\binom{n-2}{|S|-1}}\, v(S)^2 + \sum_{\substack{S\subseteq N\\ 0<|S|<n\\ i\notin S}} \frac{|S|}{n\,(n-|S|)\binom{n-2}{|S|-1}}\, v(S)^2 \\ &= \sum_{s=1}^{n-1}\left(\sum_{\substack{S\subseteq N\\ s=|S|\\ i\in S}} \frac{n-s}{sn\binom{n-2}{s-1}} + \sum_{\substack{S\subseteq N\\ s=|S|\\ i\notin S}} \frac{s}{n\,(n-s)\binom{n-2}{s-1}}\right) s^2 \\ &= \sum_{s=1}^{n-1}\left(\binom{n-1}{s-1}\frac{n-s}{sn\binom{n-2}{s-1}} + \binom{n-1}{s}\frac{s}{n\,(n-s)\binom{n-2}{s-1}}\right) s^2 \\ &= \sum_{s=1}^{n-1}\Bigg(\underbrace{\frac{n-1}{n-s}}_{>0}\underbrace{\frac{n-s}{sn}}_{>0} + \underbrace{\frac{n-1}{s}}_{>0}\underbrace{\frac{s}{n\,(n-s)}}_{>0}\Bigg)\underbrace{s^2}_{>0} > 0.\end{aligned}$$

Dividing by τ, we obtain

$$\operatorname{Var}[\hat{\phi}_i^{\mathrm{LSS}}] > 0 \quad (\forall i \in N). \quad \square$$

Lemma 2. *On the game defined by (52) with $n \geq 2$, the variance of the S-LSS estimator is zero for all players, i.e.,*

$$\operatorname{Var}[\hat{\phi}_i^{\text{S-LSS}}] = 0 \quad (\forall i \in N),$$

as long as $\tau_{j,s} > 0$ for all $j \in N$ and $s \in \{1, \ldots, n-1\}$.

Proof. Without loss of generality, we fix $j \in N$ and $s \in \{1, \ldots, n-1\}$ in order to simplify notations.

To calculate the variances of the individual stratum estimators via (36), the exact values of the stratum estimators $a_{j,s}$ are required. By using the identity (37), these are given by

$$\begin{aligned}a_{j,s} &= \sum_{\substack{S\subseteq N\\ s=|S|\\ j\in S}} m(s)\, v(S) = \frac{1}{(n-1)\binom{n-2}{s-1}} \sum_{\substack{S\subseteq N\\ s=|S|\\ j\in S}} v(S) = \frac{1}{(n-1)\binom{n-2}{s-1}}\binom{n-1}{s-1} s \\ &= \frac{n-1}{(n-1)(n-s)}\, s = \frac{s}{n-s}. \end{aligned} \tag{53}$$

Furthermore, by using (37), the sum of squared evaluations of v in (36) is given by

$$\begin{aligned}\frac{1}{(n-s)(n-1)\binom{n-2}{s-1}} \sum_{\substack{S\subseteq N\\ s=|S|\\ j\in S}} v(S)^2 &= \frac{1}{(n-s)(n-1)\binom{n-2}{s-1}}\binom{n-1}{s-1} s^2 \\ &= \frac{n-1}{(n-1)(n-s)^2}\, s^2 = \frac{s^2}{(n-s)^2}. \end{aligned} \tag{54}$$

Inserting (53) and (54) into (36) results in

$$\operatorname{Var}[\hat{a}_{j,s}] = \frac{1}{\tau_{j,s}}\left(\frac{s^2}{(n-s)^2} - a_{j,s}^2\right) = \frac{1}{\tau_{j,s}}\left(\frac{s^2}{(n-s)^2} - \frac{s^2}{(n-s)^2}\right) = 0,$$

as long as $\tau_{j,s} > 0$. Then, one can easily verify via (38) that

$$\operatorname{Var}[\hat{\phi}_i^{\text{S-LSS}}] = 0 \quad (\forall i \in N). \quad \square$$

 doi:10.20944/preprints202601.2308.v1

Corollary 1. *There exist games for which the variances of all players' S-LSS estimators are smaller than those the corresponding LSS estimators, i.e.,*

$$\mathrm{Var}[\hat{\phi}_i^{\text{S-LSS}}] < \mathrm{Var}[\hat{\phi}_i^{\text{LSS}}] \quad (\forall i \in N),$$

as long as the sample allocation scheme of S-LSS satisfies $\tau_{j,s} > 0$ for all $j \in N$ and $s \in \{1, \ldots, n-1\}$.

Proof. The proof is straightforward and follows directly from Lemmas 1 and 2. □

Variance comparison in the presence of variance within strata

Let us now consider a weighted voting game as defined in (6) with

$$\mathbf{c} = [1,2,3]^\top \quad \text{and} \quad C = 5 \tag{55}$$

as a second example. Here, we use concrete numbers in order to simplify calculations. In detail, we are interested in the variances of the Shapley value estimators of player 2 with Shapley value $\phi_2 = \frac{1}{2}$.

To simplify our later reasoning, we first calculate the population and estimator variances for all strata of the S-LSS estimator in Table 1.

s	j	elements	population var.	estimator var.
1	1	$v(\{1\}) = 0$	0	0
1	2	$v(\{2\}) = 0$	0	0
1	3	$v(\{3\}) = 0$	0	0
2	1	$v(\{1,2\}) = 0, v(\{1,3\}) = 0$	0	0
2	2	$v(\{1,2\}) = 0, v(\{2,3\}) = 1$	1/4	$1/(4\tau_{2,2})$
2	3	$v(\{1,3\}) = 0, v(\{2,3\}) = 1$	1/4	$1/(4\tau_{3,2})$

Table 1. Population and estimator variances for all strata of the S-LSS estimator.

Lemma 3. *On the game defined by (55), the variance of player 2's LSS estimator is given by*

$$\mathrm{Var}[\hat{\phi}_2^{\text{LSS}}] = \frac{5}{36\tau}.$$

Proof. The proof is straightforward by inserting (55) into (32). □

Lemma 4. *On the game defined by (55), the variance of player 2's S-LSS estimator with equal sample allocation is given by*

$$\mathrm{Var}[\hat{\phi}_2^{\text{S-LSS}}] = \frac{5}{6\tau}.$$

Proof. The proof is straightforward and is done by calculating (36), first, to obtain values for all $\mathrm{Var}[\hat{a}_{j,s}]$. Note that these $\mathrm{Var}[\hat{a}_{j,s}]$ are given in the "estimator variances" column from Table 1. Then, one obtains the final estimator's variance by inserting all $\mathrm{Var}[\hat{a}_{j,s}]$ into (38). □

Corollary 2. *There exist games for which the variance of a player's S-LSS estimator with equal sample allocation is larger than that of the LSS estimator, i.e., $\mathrm{Var}[\hat{\phi}_i^{\text{S-LSS}}] > \mathrm{Var}[\hat{\phi}_i^{\text{LSS}}]$ for some $i \in N$.*

Proof. The proof is straightforward and directly follows from Lemmas 3 and 4. □

Although an equal sample allocation scheme might be a reasonable choice, especially when calculating all players' Shapley values, one might still ask whether alternative allocation procedures could further reduce the variance of the S-LSS estimator. While our goal is to treat characteristic

 doi:10.20944/preprints202601.2308.v1

functions — and, in the context of explainable machine learning, the machine learning models behind them — as black boxes without relying on internal knowledge, let us examine a sample allocation strategy that makes use of a priori information about the variances within strata. Interestingly, we will see that even this variance-informed allocation performs worse than the LSS method on the weighted voting game defined by (55).

Lemma 5. *On the game defined by* (55)*, the variance of player* 2*'s S-LSS estimator with an optimum sample allocation, i.e., a sample allocation that is tailored to player* 2 *in order to minimize the variance of their estimator, is given by*

$$\mathrm{Var}[\hat{\phi}_2^{\text{S-LSS}}] = \frac{1}{4\tau}.$$

Proof. By inserting the estimator variances from Table 1 into (38), we obtain

$$\begin{aligned}\mathrm{Var}[\hat{\phi}_2^{\text{S-LSS}}] &= \frac{4}{9}\frac{1}{4\tau_{2,2}} + \frac{1}{9}\frac{1}{4\tau_{3,2}} = \frac{1}{9}\left(\frac{1}{\tau_{2,2}} + \frac{1}{4\tau_{3,2}}\right) \\ &= \frac{1}{9}\left(\frac{1}{x\tau} + \frac{1}{4\,(1-x)\,\tau}\right) = \frac{1}{9\tau}\left(\frac{1}{x} + \frac{1}{4\,(1-x)}\right),\end{aligned} \tag{56}$$

whereby $x \in (0,1)$ is the fraction of τ that is used for estimating the stratum with $j = 2, s = 2$ and $(1-x)$ is the fraction of τ that is used for estimating the stratum with $j = 3, s = 2$.

Therefore, we get the proxy minimization problem

$$\min_{x\in(0,1)} \frac{1}{x} + \frac{1}{4\,(1-x)}.$$

Let us now set

$$f(x) = \frac{1}{x} + \frac{1}{4\,(1-x)}$$

with its derivative given by

$$f'(x) = -\frac{1}{x^2} + \frac{1}{4\,(1-x)^2}.$$

Setting $f'(x) = 0$, we get

$$x = \frac{4 \pm 2}{3},$$

with $x = \frac{2}{3}$ being the only solution in $(0,1)$. Furthermore, since

$$f''(x) = \frac{2}{x^3} + \frac{1}{2\,(1-x)^3} > 0 \quad \text{on } (0,1),$$

we obtain that $x = \frac{2}{3}$ indeed is a minimum with value $f\left(\frac{2}{3}\right) = \frac{9}{4}$. Thus, via (56), the minimum variance of the S-LSS estimator is given by

$$\mathrm{Var}[\hat{\phi}_2^{\text{S-LSS}}] = \frac{1}{9\tau}\frac{9}{4} = \frac{1}{4\tau}. \quad \square$$

Corollary 3. *There exist games for which the variance of a player's S-LSS estimator with optimum sample allocation, i.e., a sample allocation that is tailored to that specific player in order to minimize the variance of their estimator, is larger than that of the LSS estimator, i.e.,* $\mathrm{Var}[\hat{\phi}_i^{\text{S-LSS}}] > \mathrm{Var}[\hat{\phi}_i^{\text{LSS}}]$ *for some* $i \in N$.

Proof. The proof is straightforward and directly follows from Lemmas 3 and 5. $\square$

Concluding Variance Comparison of LSS and S-LSS

Recapitulating the results from the previous two paragraphs, we obtain:

 doi:10.20944/preprints202601.2308.v1

Theorem 2. *Depending on the specific cooperative game, the variance of a player's Shapley value estimator may be smaller under either LSS or S-LSS, even when the S-LSS algorithm uses an optimal, player-specific sample allocation.*

Proof. The proof is straightforward and directly follows from Corollaries 1 and 3. □

To summarize the results in this subsection so far, Proposition 5 demonstrates that S-LSS is not the stratified variant of LSS in a sense of the definition provided in Section 3.2, and Theorem 2 states that neither of both algorithms is preferable over the other one on all cooperative games. Since the proof of Theorem 2 builds on the calculation of both algorithms' theoretical variances on two distinct games, it does not provide an intuitive explanation why S-LSS may perform worse than LSS in some settings. Therefore, we provide an explanation in the following.

Since Benati et al. [14] did not provide a sample allocation scheme for the S-LSS algorithm, we first derive a proportional sample allocation similar to Subsection 3.2, which is motivated by the fact that all coalition sizes are equally likely in the context of LSS, i.e.,

$$\mathbb{P}(s = |\mathcal{S}|) = \frac{1}{n-1} \quad (\forall s \in \{1, \ldots, n-1\}),$$

with $\mathcal{S} \sim p^{\text{LSS}}$, whereby p^{LSS} is the probability mass function from (28).

Furthermore, once a random coalition $\mathcal{S} \sim p^{\text{LSS}}$ of some size $s \in \{1, \ldots, n-1\}$ is sampled, the probability of any player being in it is the same for all players, i.e.,

$$\mathbb{P}(j \in \mathcal{S} \mid s = |\mathcal{S}|) = \frac{s}{n} \quad (\forall j \in N).$$

Then, the overall probability that a random coalition $\mathcal{S} \sim p^{\text{LSS}}$ belongs to a specific stratum of the S-LSS algorithm — which is the case when $\mathcal{S}$ meets a specific size s and, additionally, some specific player j is in it — is given by

$$\mathbb{P}(s = |\mathcal{S}|)\,\mathbb{P}(j \in \mathcal{S} \mid s = |\mathcal{S}|) = \frac{s}{n\,(n-1)} \quad (\forall j \in N, \forall s \in \{1, \ldots, n-1\}). \tag{57}$$

As we established at the beginning of this subsection, the statement on variance reduction for a stratified estimator with stratum sample sizes proportional to the probabilities of sampled elements belonging to a stratum from Subsection 3.2 does not apply in this context since S-LSS is not a valid stratified variant of LSS. However, the proportional sample allocation will come out as a useful starting point for our subsequent explanations. Moreover, as established in Theorem 2, the exact choice of the applied sample allocation scheme does not alter the fact that S-LSS can perform worse than LSS, since the increased variance of S-LSS was also observed under an optimal allocation scheme. Additionally, as we will show later, sample allocation strategies other than the proportional sample allocation do not change the outcome and conclusions of our following reasoning.

Hence we set all $\tau_{j,s}^{\text{S-LSS}} \propto \mathbb{P}(s = |\mathcal{S}|)\,\mathbb{P}(j \in \mathcal{S} \mid s = |\mathcal{S}|)$. Then, we obtain

$$\begin{aligned}
\tau_{j,s}^{\text{S-LSS}} &= \tau \frac{\mathbb{P}(s = |\mathcal{S}|)\,\mathbb{P}(j \in \mathcal{S} \mid s = |\mathcal{S}|)}{\sum_{s'=1}^{n-1} \sum_{j' \in N} \mathbb{P}(s' = |\mathcal{S}|)\,\mathbb{P}(j' \in \mathcal{S} \mid s' = |\mathcal{S}|)} \\
&= \frac{s\tau}{n\,(n-1) \sum_{s'=1}^{n-1} \sum_{j' \in N} \frac{s'}{n\,(n-1)}} = \frac{s\tau}{\sum_{s'=1}^{n-1} \sum_{j' \in N} s'} = \frac{s\tau}{n \sum_{s'=1}^{n-1} s'} \\
&= \frac{2s\tau}{n^2\,(n-1)} \quad (\forall j \in N, \forall s \in \{1, \ldots, n-1\}).
\end{aligned} \tag{58}$$

 doi:10.20944/preprints202601.2308.v1

On the other hand, let us now find out how often a sample is expected to belong to a stratum $a_{j,s}$ of S-LSS when actually running LSS, which is denoted by $\mathbb{E}[\tau_{j,s}^{\text{LSS}}]$. Via (57), we obtain

$$\begin{aligned}\mathbb{E}[\tau_{j,s}^{\text{LSS}}] &= \tau\,\mathbb{P}(s = |\mathcal{S}|)\,\mathbb{P}(j \in \mathcal{S} \mid s = |\mathcal{S}|) \\ &= \frac{s\tau}{n\,(n-1)} \quad (\forall j \in N, \forall s \in \{1,\dots,n-1\}).\end{aligned} \tag{59}$$

Finally, by comparing (58) and (59), we observe

$$\tau_{j,s}^{\text{S-LSS}} = \frac{2s\tau}{n^2\,(n-1)} \overset{n>2}{<} \frac{s\tau}{n\,(n-1)} = \mathbb{E}[\tau_{j,s}^{\text{LSS}}] \tag{60}$$

for all $j \in N$ and $s \in \{1,\dots,n-1\}$. Clearly, (60) shows that the expected sample size for stratum $a_{j,s}$ is larger by a factor of $\frac{n}{2}$ when executing LSS in comparison to the corresponding sample size of S-LSS with a proportional sample allocation. This is due to the fact that LSS actually updates multiple strata with one evaluation of v, see the beginning of this subsection. We conclude that S-LSS might eliminate the variance between the estimators of different strata (similar to the definition provided in Subsection 3.2), but taking into account the results from Proposition 3, one obtains the dependency on $\tau_{j,s}^{\text{S-LSS}}$ when quantifying the stratum estimators' variances, which results in larger variances of the individual stratum estimators due to smaller $\tau_{j,s}^{\text{S-LSS}}$.

This underscores our observations from the previous paragraphs. Whenever the variances within strata are small, e.g., in the context of the game defined by (52), the small sample sizes for each stratum are enough for S-LSS to provide accurate stratum estimators, and the variance between the strata is eliminated by algorithm design. Contrary, if the variances within strata are larger, e.g., in the context of the game defined by (55), the reduced amount of samples available for individual strata of the S-LSS algorithm might increase the stratum estimators' variances more than the stratified algorithm design reduces the overall variance.

This result does not only apply to our chosen sample allocation scheme, i.e., the proportional sample allocation. Clearly, from (60), one obtains that there are too little samples available for the individual stratum estimators in comparison to LSS, and any other sample allocation scheme would also have the drawback that there exists at least one stratum for which there are fewer samples available in comparison to LSS. Additionally, we highlight that even an optimized sample allocation cannot account for the reduced amount of samples available, see Lemma 5 and Corollary 3.

In summary, we recommend using LSS whenever one expects large variances within the strata of S-LSS and a small variance between the stratum estimators. On the other hand, whenever one expects a large variance between the stratum estimators and small variances within strata, we recommend choosing S-LSS. In the latter case, if one has information about the exact variances of individual strata or a good approximation of them, one could distribute the sample sizes across strata similar to the procedure from Lemma 5. Additionally, we highlight that several improvements can be made in order to reduce the mutual shortcomings of both algorithms. One approach is the reuse of samples across strata in order to eliminate the sample size inequality from (60), which was proposed by Benati et al. [14] and introduced as SRS-LSS in Subsection 5.2. However, as we will point out in Subsection 5.4, this approach adds covariance terms to the overall variances of the Shapley value estimators. Another improvement could be some kind of two-stage algorithm, where the first stage is used to decide whether the LSS or S-LSS algorithm should be executed in the second stage.

Finally, we visualize the behavior of LSS and S-LSS when applied to different cooperative games. Figure 2 illustrates the previously obtained results by plotting the variances of LSS and S-LSS against weighted voting games with different quotas. As expected, the variances show a clear dependence to the underlying game, determining whether LSS or S-LSS performs better.

 doi:10.20944/preprints202601.2308.v1

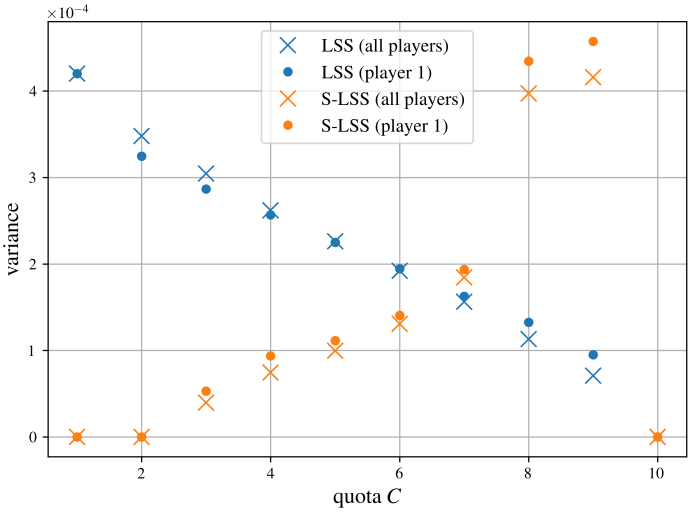

Figure 2. Theoretical variance comparison of LSS and S-LSS evaluated on the weighted voting game defined by (7). The sample budget is $\tau = 10000$, and the sample sizes of S-LSS are distributed according to (58) with ceiling operations applied whenever the result is not an integer. The crosses denote the mean variance across all players' Shapley values, while the dots specify the variance of $\hat{\phi}_1$.

5.4. Analysis of the Variance of SRS-LSS

Since Benati et al. [14] did not address the variance of the SRS-LSS algorithm, we derive:

Proposition 6. *For all $i \in N$, the variance of the SRS-LSS estimator is given by*

$$\begin{aligned}\operatorname{Var}[\hat{\phi}_i] = &\left(\frac{n-1}{n}\right)^2 \sum_{s=1}^{n-1} \operatorname{Var}[\hat{a}_{i,s}] \\ &+ \frac{1}{n^2}\left(\sum_{\substack{j\in N\\ j\neq i}} \sum_{s=1}^{n-1} \operatorname{Var}[\hat{a}_{j,s}] + 2 \sum_{\substack{j,k\in N\\ j<k\\ j,k\neq i}} \sum_{s=1}^{n-1} \operatorname{Cov}[\hat{a}_{j,s}, \hat{a}_{k,s}]\right) \\ &- 2\,\frac{n-1}{n^2} \sum_{\substack{j\in N\\ j\neq i}} \sum_{s=1}^{n-1} \operatorname{Cov}[\hat{a}_{i,s}, \hat{a}_{j,s}],\end{aligned} \tag{61}$$

with $\operatorname{Var}[\hat{a}_{j,s}]$ *given in* (36),

$$\operatorname{Cov}[\hat{a}_{j,s}, \hat{a}_{k,s}] = \frac{1}{(n-s)^2} \operatorname{Cov}\left[\frac{1}{\hat{\tau}_{j,s}} \sum_{\mathcal{S}\in\mathbb{S}_s} \mathbb{1}_{j\in\mathcal{S}}\, v(\mathcal{S}), \frac{1}{\hat{\tau}_{k,s}} \sum_{\mathcal{S}\in\mathbb{S}_s} \mathbb{1}_{k\in\mathcal{S}}\, v(\mathcal{S})\right], \tag{62}$$

$$\hat{\tau}_{j,s} = \sum_{\mathcal{S}\in\mathbb{S}_s} \mathbb{1}_{j\in\mathcal{S}} \quad \textit{and} \quad \hat{\tau}_{k,s} = \sum_{\mathcal{S}\in\mathbb{S}_s} \mathbb{1}_{k\in\mathcal{S}},$$

and $\mathbb{S}_s \sim_{\text{iid}} \mathcal{U}(\{S \subseteq N \mid s = |S|\})$.

Proof. Recall the closed-form formulation of the Shapley value as a member of the family of least square values in (27). By using the stratification scheme introduced in (34), one obtains the estimator

$$\hat{\phi}_i = \frac{v(N)}{n} + \frac{n-1}{n}\,\hat{a}_i - \frac{1}{n} \sum_{\substack{j\in N\\ j\neq i}} \hat{a}_j$$

 doi:10.20944/preprints202601.2308.v1

with

$$\hat{a}_j = \sum_{s=1}^{n-1} \hat{a}_{j,s}. \tag{63}$$

With these definitions in place, the variance of the Shapley value estimator is given by

$$\begin{aligned}\mathrm{Var}[\hat{\phi}_i] &= \mathrm{Var}\left[\frac{v(N)}{n} + \frac{n-1}{n}\,\hat{a}_i - \frac{1}{n}\sum_{\substack{j\in N\\ j\neq i}} \hat{a}_j\right] \\ &= \left(\frac{n-1}{n}\right)^2 \mathrm{Var}[\hat{a}_i] + \frac{1}{n^2}\,\mathrm{Var}\left[\sum_{\substack{j\in N\\ j\neq i}} \hat{a}_j\right] - 2\,\frac{n-1}{n^2}\,\mathrm{Cov}\left[\hat{a}_i, \sum_{\substack{j\in N\\ j\neq i}} \hat{a}_j\right],\end{aligned} \tag{64}$$

where

$$\mathrm{Var}\left[\sum_{\substack{j\in N\\ j\neq i}} \hat{a}_j\right] = \sum_{\substack{j\in N\\ j\neq i}} \mathrm{Var}[\hat{a}_j] + 2 \sum_{\substack{j,k\in N\\ j<k\\ j,k\neq i}} \mathrm{Cov}[\hat{a}_j, \hat{a}_k], \tag{65}$$

and

$$\mathrm{Cov}\left[\hat{a}_i, \sum_{\substack{j\in N\\ j\neq i}} \hat{a}_j\right] = \sum_{\substack{j\in N\\ j\neq i}} \mathrm{Cov}[\hat{a}_i, \hat{a}_j]. \tag{66}$$

Then, via (63), we rewrite

$$\mathrm{Cov}[\hat{a}_j, \hat{a}_k] = \mathrm{Cov}\left[\sum_{s=1}^{n-1} \hat{a}_{j,s}, \sum_{s'=1}^{n-1} \hat{a}_{k,s'}\right] = \sum_{s=1}^{n-1}\sum_{s'=1}^{n-1} \mathrm{Cov}[\hat{a}_{j,s}, \hat{a}_{k,s'}],$$

for all $j \neq k \in N$, but since samples are independent for $s \neq s'$, the equation can be simplified to

$$\mathrm{Cov}[\hat{a}_j, \hat{a}_k] = \sum_{s=1}^{n-1} \mathrm{Cov}[\hat{a}_{j,s}, \hat{a}_{k,s}].$$

Next, from line 12 in Algorithm 3, we derive

$$\begin{aligned}\mathrm{Cov}[\hat{a}_{j,s}, \hat{a}_{k,s}] &= \mathrm{Cov}\left[\frac{1}{\hat{\tau}_{j,s}\,(n-s)} \sum_{\mathcal{S}\in\mathbb{S}_s} \mathbb{1}_{j\in\mathcal{S}}\, v(\mathcal{S}), \frac{1}{\hat{\tau}_{k,s}\,(n-s)} \sum_{\mathcal{S}\in\mathbb{S}_s} \mathbb{1}_{k\in\mathcal{S}}\, v(\mathcal{S})\right] \\ &= \frac{1}{(n-s)^2}\,\mathrm{Cov}\left[\frac{1}{\hat{\tau}_{j,s}} \sum_{\mathcal{S}\in\mathbb{S}_s} \mathbb{1}_{j\in\mathcal{S}}\, v(\mathcal{S}), \frac{1}{\hat{\tau}_{k,s}} \sum_{\mathcal{S}\in\mathbb{S}_s} \mathbb{1}_{k\in\mathcal{S}}\, v(\mathcal{S})\right]\end{aligned}$$

with $\hat{\tau}_{j,s} = \sum_{\mathcal{S}\in\mathbb{S}_s} \mathbb{1}_{j\in\mathcal{S}}$, $\hat{\tau}_{k,s} = \sum_{\mathcal{S}\in\mathbb{S}_s} \mathbb{1}_{k\in\mathcal{S}}$, and $\mathbb{S}_s \sim_{\text{iid}} \mathcal{U}\big(\{S \subseteq N \mid s = |S|\}\big)$.

Finally, setting

$$\mathrm{Var}[\hat{a}_j] = \sum_{s=1}^{n-1} \mathrm{Var}[\hat{a}_{j,s}],$$

since $\mathrm{Cov}[\hat{a}_{j,s}, \hat{a}_{j,s'}] = 0$ for $s \neq s'$ and inserting everything back into (64) results in (61). □

Corollary 4. *For S-LSS,* (61) *is equal to* (38).

Proof. In the case of S-LSS, sampling within each stratum is independent of sampling in all other strata. Therefore, the covariance terms defined by (62) are zero. Consequently, setting these covariance terms to zero in (61) results in (38). □

Theorem 3. *For SRS-LSS, the covariance terms in* (61) *do not vanish in general.*

 doi:10.20944/preprints202601.2308.v1

Proof. To simplify notations, we first obtain from (62) that the constant, non-zero term $(n-s)^{-2} > 0$ can be ignored when demonstrating that $\mathrm{Cov}[\hat{a}_{j,s}, \hat{a}_{k,s}] \neq 0$ might occur. Therefore, we define

$$\mathrm{Cov}[\bar{a}_{j,s}, \bar{a}_{k,s}] = \mathrm{Cov}\left[\frac{1}{\hat{\tau}_{j,s}} \sum_{\mathcal{S}\in\mathbb{S}_s} \mathbb{1}_{j\in\mathcal{S}}\, v(\mathcal{S}), \frac{1}{\hat{\tau}_{k,s}} \sum_{\mathcal{S}\in\mathbb{S}_s} \mathbb{1}_{k\in\mathcal{S}}\, v(\mathcal{S})\right].$$

Now, we demonstrate by example that $\mathrm{Cov}[\bar{a}_{j,s}, \bar{a}_{k,s}]$ may be non-zero. For this purpose, we define a weighted voting game with $\mathbf{c} = [1,0,1]$ and quota $C = 2$, such that the only winning coalition of size 2 is $\{1,3\}$. Let us now fix $s = 2$, such that the corresponding sample is obtained as $\mathbb{S}_2 \sim_{\text{iid}} \mathcal{U}(\{\{1,2\},\{1,3\},\{2,3\}\})$. Without loss of generality, we set the sample size to $\tau_2 = 3$.

In Proposition 4, we established that the probability of at least one value $\hat{\tau}_{j,s}$ being zero converges to zero as $\tau_s \to \infty$. For $\tau_2 = 3$, this probability is given by

$$\mathbb{P}(\mathcal{S}_1 = \mathcal{S}_2 = \mathcal{S}_3) = 3\left(\frac{1}{3}\right)^3 = \frac{1}{9}, \tag{67}$$

where $\mathbb{S}_2 = [\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3]^\top$. We denote this event by B and its complement by $\overline{B}$, such that $\mathbb{P}(B) = \mathbb{P}(\mathcal{S}_1 = \mathcal{S}_2 = \mathcal{S}_3)$.

Although in this example the probability of failing to sample at least one element from each stratum is relatively large, we proceed under the standard assumption that this probability is negligible, which is justified for large τ_2. Therefore, we calculate the conditional expectations in Table 2 based on the assumption that B does not occur.

Table 2. All valid realizations of the sample $\mathbb{S}_2$ (order ignored) when $\tau_2 = 3$, together with their respective probabilities as well the corresponding resulting values of $\bar{a}_{1,2}$, $\bar{a}_{3,2}$, and $\bar{a}_{1,2}\,\bar{a}_{3,2}$. The last row shows the expected values over all valid realizations, i.e., conditioned on $\overline{B}$.

#permutations	$\mathbb{P}(\mathbb{S}_2)$	$\mathbb{P}(\mathbb{S}_2 \mid \overline{B})$	elements of $\mathbb{S}_2$	$\bar{a}_{1,2}$	$\bar{a}_{3,2}$	$\bar{a}_{1,2}\,\bar{a}_{3,2}$
6/27	2/9	2/8	$\{1,2\},\{1,3\},\{2,3\}$	1/2	1/2	1/4
3/27	1/9	1/8	$\{1,2\},\{1,2\},\{1,3\}$	1/3	1	1/3
3/27	1/9	1/8	$\{1,2\},\{1,2\},\{2,3\}$	0	0	0
3/27	1/9	1/8	$\{1,2\},\{1,3\},\{1,3\}$	2/3	1	2/3
3/27	1/9	1/8	$\{1,2\},\{2,3\},\{2,3\}$	0	0	0
3/27	1/9	1/8	$\{1,3\},\{1,3\},\{2,3\}$	1	2/3	2/3
3/27	1/9	1/8	$\{1,3\},\{2,3\},\{2,3\}$	1	1/3	1/3
$\mathbb{E}[\,\cdot \mid \overline{B}]$				1/2	1/2	5/16

Based on Table 2, the conditional covariance between $\bar{a}_{1,2}$ and $\bar{a}_{3,2}$ can be calculated as

$$\mathrm{Cov}[\bar{a}_{1,2}, \bar{a}_{3,2} \mid \overline{B}] = \mathbb{E}[\bar{a}_{1,2}\,\bar{a}_{3,2} \mid \overline{B}] - \mathbb{E}[\bar{a}_{1,2} \mid \overline{B}]\,\mathbb{E}[\bar{a}_{3,2} \mid \overline{B}] = \frac{5}{16} - \frac{1}{2}\frac{1}{2} = \frac{1}{16} > 0.$$

Although conditioning on $\overline{B}$ is necessary for computing the covariance, it may slightly affect its numerical value. The true (unconditional) covariance could differ marginally. Accounting for the undefined cases would require specifying a particular strategy to handle such edge cases, which was not proposed for the SRS-LSS algorithm and lies beyond the scope of this investigation. Nevertheless, the key point remains that the covariance does not vanish. As can be observed from the $\bar{a}_{1,2}$ and $\bar{a}_{3,2}$ columns in Table 2, these quantities have a clear positive correlation. Since players 1 and 3 together form the only winning coalition of size 2, large realizations of $\bar{a}_{1,2}$ necessarily coincide with large realizations of $\bar{a}_{3,2}$. This dependence confirms that $\mathrm{Cov}[\bar{a}_{1,2}, \bar{a}_{3,2}] > 0$, and as a result, we have $\mathrm{Cov}[\hat{a}_{1,2}, \hat{a}_{3,2}] \neq 0$. □

 doi:10.20944/preprints202601.2308.v1

From Theorem 3, it follows that reusing evaluations of the characteristic function introduces non-zero covariance terms. These terms are hard to characterize, which complicates the exact computation of the SRS-LSS estimator's theoretical variance and its comparison to LSS and S-LSS. A detailed analysis of (62) together with the resulting influence on (61) might be a question for further research. In the meantime, we rely on empirical observations of the SRS-LSS estimator's variance in the numerical experiments in Section 6, which indicate a significantly reduced variance of SRS-LSS in comparison to LSS and S-LSS.

5.5. Comparison of LSS and UKS

In the following, we compare LSS (Algorithm 1) with UKS (see the end of Subsection 5.2). First, we show how UKS fits into the general importance sampling framework defined by Theorem 1 and discuss the relationship between both algorithms within this context. Subsequently, we compare their variances.

Before going into detail, we note that the formulation of UKS derived in the following was already proposed by Fumagalli et al. [30]. The authors identify UKS as a special case of their *SHAPley Interaction Quantification* framework, which enables a reformulation of the algorithm similar to ours. Nevertheless, we present our derivations for several reasons. First, our derivation of UKS is, from our perspective, more intuitive. Second, while Fumagalli et al. [30] suggest that their framework enables a particular formulation of UKS that arises specifically from their approach, this form of UKS can also be directly obtained by expanding the matrices and vectors in (46) without using the indirection via their *SHAPley Interaction Quantification* framework. Third, UKS has not yet been formally integrated into the importance sampling framework defined by Theorem 1, which allows us to directly apply our results from Proposition 1. Finally, and most importantly, to the best of our knowledge, we are the first to demonstrate that UKS approximates the same underlying problem as LSS, establishing a novel link between the two algorithms.

We refer to Appendix A, which shows that our following results are consistent with those of Fumagalli et al. [30].

Algorithm Comparison

First, recall the definition of LSS given in (30) with w_i being defined in (24), m in (25), $\gamma = 1$ in (26), and p in (28). Since we have only derived $w_i(S)/p(S)$ so far and not $w_i(S)$ directly, inserting m into w_i results in

$$\begin{aligned}\frac{n-|S|}{n\,(n-1)\binom{n-2}{|S|-1}} &= \frac{(n-|S|)\,(|S|-1)!\,(n-|S|-1)!}{n\,(n-1)\,(n-2)!} = \frac{(|S|-1)!\,(n-|S|)!}{n!} \\ &= \frac{1}{|S|}\frac{|S|!\,(n-|S|)!}{n!} = \frac{1}{|S|\binom{n}{|S|}}\end{aligned}$$

for the case $i \in S$, and

$$\begin{aligned}-\frac{|S|}{n\,(n-1)\binom{n-2}{|S|-1}} &= -\frac{|S|\,(|S|-1)!\,(n-|S|-1)!}{n\,(n-1)\,(n-2)!} = -\frac{|S|!\,(n-|S|-1)!}{n!} \\ &= -\frac{1}{n-|S|}\frac{|S|!\,(n-|S|)!}{n!} = -\frac{1}{(n-|S|)\binom{n}{|S|}}\end{aligned}$$

for the case $i \notin S$. Thus, w_i can be rewritten as

$$w_i^{\mathrm{LSS}}(S) = \begin{cases} \frac{1}{|S|\binom{n}{|S|}} & \text{if } i \in S \\ -\frac{1}{(n-|S|)\binom{n}{|S|}} & \text{if } i \notin S. \end{cases} \tag{68}$$

Now, let us turn our attention to the UKS algorithm:

 doi:10.20944/preprints202601.2308.v1

Proposition 7. *UKS can be brought into the form*

$$\hat{\phi}_i = t + \frac{1}{\tau} \sum_{\mathcal{S} \in \mathbb{S}^{\mathrm{UKS}}} \frac{w_i^{\mathrm{UKS}}(\mathcal{S})}{p^{\mathrm{UKS}}(\mathcal{S})}\, v(\mathcal{S})$$

with $\mathbb{S}^{\mathrm{UKS}} \sim_{\mathrm{iid}} p^{\mathrm{UKS}}$ *being a sample according to* (41),

$$t = \left[A^{-1}\mathbf{1} \frac{v(N)}{\mathbf{1}^\top A^{-1}\mathbf{1}} \right]_i$$

being a constant, and

$$\frac{w_i^{\mathrm{UKS}}(S)}{p^{\mathrm{UKS}}(S)} = \left[A^{-1}\,\mathbf{z}(S)\right]_i - \left[A^{-1}\mathbf{1}\right]_i \frac{\mathbf{1}^\top A^{-1}\,\mathbf{z}(S)}{\mathbf{1}^\top A^{-1}\mathbf{1}}. \tag{69}$$

Proof. Starting from (46) and substituting $\hat{\boldsymbol{b}}$ as defined in (45) results in

$$\begin{aligned}
\hat{\phi}_i &= \left[A^{-1}\left(\hat{\boldsymbol{b}} - \mathbf{1} \frac{\mathbf{1}^\top A^{-1}\hat{\boldsymbol{b}} - v(N)}{\mathbf{1}^\top A^{-1}\mathbf{1}} \right) \right]_i \\
&= \left[A^{-1}\hat{\boldsymbol{b}} - A^{-1}\mathbf{1} \frac{\mathbf{1}^\top A^{-1}\hat{\boldsymbol{b}}}{\mathbf{1}^\top A^{-1}\mathbf{1}} + A^{-1}\mathbf{1} \frac{v(N)}{\mathbf{1}^\top A^{-1}\mathbf{1}} \right]_i \\
&= \underbrace{\left[A^{-1}\mathbf{1} \frac{v(N)}{\mathbf{1}^\top A^{-1}\mathbf{1}} \right]_i}_{t} + \left[A^{-1}\hat{\boldsymbol{b}} - A^{-1}\mathbf{1} \frac{\mathbf{1}^\top A^{-1}}{\mathbf{1}^\top A^{-1}\mathbf{1}}\, \hat{\boldsymbol{b}} \right]_i \\
&= t + \left[\left(A^{-1} - A^{-1}\mathbf{1} \frac{\mathbf{1}^\top A^{-1}}{\mathbf{1}^\top A^{-1}\mathbf{1}} \right) \hat{\boldsymbol{b}} \right]_i \\
&= t + \left[\left(A^{-1} - A^{-1}\mathbf{1} \frac{\mathbf{1}^\top A^{-1}}{\mathbf{1}^\top A^{-1}\mathbf{1}} \right) \frac{1}{\tau} \sum_{\mathcal{S} \in \mathbb{S}^{\mathrm{UKS}}} \mathbf{z}(\mathcal{S})\, v(\mathcal{S}) \right]_i \\
&= t + \left[\frac{1}{\tau} \sum_{\mathcal{S} \in \mathbb{S}^{\mathrm{UKS}}} \left(A^{-1} - A^{-1}\mathbf{1} \frac{\mathbf{1}^\top A^{-1}}{\mathbf{1}^\top A^{-1}\mathbf{1}} \right) \mathbf{z}(\mathcal{S})\, v(\mathcal{S}) \right]_i \\
&= t + \left[\frac{1}{\tau} \sum_{\mathcal{S} \in \mathbb{S}^{\mathrm{UKS}}} \left(A^{-1}\,\mathbf{z}(\mathcal{S}) - A^{-1}\mathbf{1} \frac{\mathbf{1}^\top A^{-1}\,\mathbf{z}(\mathcal{S})}{\mathbf{1}^\top A^{-1}\mathbf{1}} \right) v(\mathcal{S}) \right]_i \\
&= t + \frac{1}{\tau} \sum_{\mathcal{S} \in \mathbb{S}^{\mathrm{UKS}}} \underbrace{\left(\left[A^{-1}\,\mathbf{z}(\mathcal{S})\right]_i - \left[A^{-1}\mathbf{1}\right]_i \frac{\mathbf{1}^\top A^{-1}\,\mathbf{z}(\mathcal{S})}{\mathbf{1}^\top A^{-1}\mathbf{1}} \right)}_{w_i^{\mathrm{UKS}}(\mathcal{S}) / p^{\mathrm{UKS}}(\mathcal{S})} v(\mathcal{S}). \quad \square
\end{aligned} \tag{70}$$

We now aim to show that both algorithms share the same weight function w and the same initial term, i.e., $w_i^{\mathrm{LSS}}(S) = w_i^{\mathrm{UKS}}(S)$ and $t = \frac{v(N)}{n}$. To prepare for this, we first carry out some reformulations in order to simplify the upcoming calculations. Readers only interested in the final result may skip ahead to Theorem 4.

Lemma 6. *The expression for* Q *given in* (43) *can be rewritten as*

$$Q = \frac{2\,(n-1)\,H_{n-1}}{n}, \tag{71}$$

where H_{n-1} *denotes the* $(n-1)$*-th harmonic number.*

Proof. The proof is straightforward by inserting the identity $\sum_{s=1}^{n-1} \frac{1}{s\,(n-s)} = \frac{2H_{n-1}}{n}$ with H_{n-1} being the $(n-1)$-th harmonic number [30] into (43), which results in (71). $\square$

 doi:10.20944/preprints202601.2308.v1

Lemma 7. *The inverse of A as defined in (47) is given by*

$$A^{-1} = \begin{bmatrix} 2H_{n-1} + \overline{\rho} & \overline{\rho} & \cdots & \overline{\rho} \\ \overline{\rho} & 2H_{n-1} + \overline{\rho} & \cdots & \overline{\rho} \\ \vdots & \vdots & \ddots & \vdots \\ \overline{\rho} & \overline{\rho} & \cdots & 2H_{n-1} + \overline{\rho} \end{bmatrix}$$

with

$$\overline{\rho} = -\frac{2H_{n-1}\left(H_{n-1} - 1\right)}{1 + n\left(H_{n-1} - 1\right)}.$$

Proof. The article by Fumagalli et al. [30] shows that the off-diagonal entries of the inverse of A as defined in (47) are given by

$$\overline{\rho} = -2H_{n-1}\,\frac{\frac{H_{n-1}-1}{2H_{n-1}}}{\frac{1}{2} + (n-1)\,\frac{H_{n-1}-1}{2H_{n-1}}}.$$

Its denominator can be reformulated as

$$\begin{aligned} \frac{1}{2} + (n-1)\,\frac{H_{n-1}-1}{2H_{n-1}} &= \frac{H_{n-1} + (n-1)\left(H_{n-1}-1\right)}{2H_{n-1}} \\ &= \frac{1 + \left(H_{n-1}-1\right) + (n-1)\left(H_{n-1}-1\right)}{2H_{n-1}} \\ &= \frac{1 + n\left(H_{n-1}-1\right)}{2H_{n-1}}, \end{aligned} \tag{72}$$

and thus,

$$\overline{\rho} = -2H_{n-1}\,\frac{\frac{H_{n-1}-1}{2H_{n-1}}}{\frac{1+n\left(H_{n-1}-1\right)}{2H_{n-1}}} = -\frac{2H_{n-1}\left(H_{n-1}-1\right)}{1 + n\left(H_{n-1}-1\right)}.$$

For the diagonal entries, Fumagalli et al. [30] provides

$$2H_{n-1}\,\frac{\frac{1}{2} + (n-2)\,\frac{H_{n-1}-1}{2H_{n-1}}}{\frac{1}{2} + (n-1)\,\frac{H_{n-1}-1}{2H_{n-1}}},$$

and similarly to (72), its numerator can be derived as

$$\frac{1}{2} + (n-2)\,\frac{H_{n-1}-1}{2H_{n-1}} = \frac{1 + (n-1)\left(H_{n-1}-1\right)}{2H_{n-1}}. \tag{73}$$

Putting (72) and (73) together, we retrieve

$$\begin{aligned} 2H_{n-1}\,\frac{\frac{1}{2} + (n-2)\,\frac{H_{n-1}-1}{2H_{n-1}}}{\frac{1}{2} + (n-1)\,\frac{H_{n-1}-1}{2H_{n-1}}} &= 2H_{n-1}\,\frac{\frac{1+(n-1)\left(H_{n-1}-1\right)}{2H_{n-1}}}{\frac{1+n\left(H_{n-1}-1\right)}{2H_{n-1}}} \\ &= 2H_{n-1}\,\frac{1 + (n-1)\left(H_{n-1}-1\right)}{1 + n\left(H_{n-1}-1\right)} \\ &= 2H_{n-1} - 2H_{n-1}\left(1 - \frac{1 + (n-1)\left(H_{n-1}-1\right)}{1 + n\left(H_{n-1}-1\right)}\right) \\ &= 2H_{n-1} - 2H_{n-1}\,\frac{2H_{n-1}(1 + n\left(H_{n-1}-1\right) - 1 - (n-1)\left(H_{n-1}-1\right))}{1 + n\left(H_{n-1}-1\right)} \\ &= 2H_{n-1} - \frac{2H_{n-1}\left(H_{n-1}-1\right)}{1 + n\left(H_{n-1}-1\right)} = 2H_{n-1} + \overline{\rho}. \quad \square \end{aligned}$$

 doi:10.20944/preprints202601.2308.v1

Proposition 8. *The constant terms of both algorithms are equal, i.e.,*

$$t = \frac{v(N)}{n}.$$

Proof. From (47) one directly obtains that $\mathbf{1}$ is an eigenvector of A with eigenvalue $\lambda = \frac{1}{2} + (n-1)\,\rho$. Thus, we have $A^{-1}\mathbf{1} = \lambda^{-1}\mathbf{1}$. With that given, rewriting t from Proposition 7 results in

$$\begin{aligned} t &= \left[A^{-1}\mathbf{1}\frac{v(N)}{\mathbf{1}^\top A^{-1}\mathbf{1}}\right]_i \\ &= \left[A^{-1}\mathbf{1}\frac{1}{\mathbf{1}^\top (A^{-1}\mathbf{1})}\right]_i v(N) = \left[\lambda^{-1}\mathbf{1}\frac{1}{\mathbf{1}^\top (\lambda^{-1}\mathbf{1})}\right]_i v(N) \\ &= \left[\lambda^{-1}\mathbf{1}\frac{1}{n\lambda^{-1}}\right]_i v(N) = \frac{\lambda^{-1}}{n\lambda^{-1}}\, v(N) = \frac{v(N)}{n}. \quad \square \end{aligned}$$

Lemma 8. *For any $S \subseteq N$ and $i \in N$, there holds*

$$\left[A^{-1}\mathbf{1}\right]_i \frac{\mathbf{1}^\top A^{-1}\,\mathbf{z}(S)}{\mathbf{1}^\top A^{-1}\mathbf{1}} = \frac{2|S|H_{n-1}}{n} + |S|\bar{\rho}.$$

Proof. Using Lemma 7, we derive

$$\mathbf{1}^\top A^{-1} = \left[2H_{n-1} + n\bar{\rho}, \ldots, 2H_{n-1} + n\bar{\rho}\right] \in \mathbb{R}^{1\times n},$$

and thus,

$$\frac{\mathbf{1}^\top A^{-1}\,\mathbf{z}(S)}{\mathbf{1}^\top A^{-1}\mathbf{1}} = \frac{|S|\,(2H_{n-1} + n\bar{\rho})}{n\,(2H_{n-1} + n\bar{\rho})} = \frac{|S|}{n}.$$

Furthermore, also based on Lemma 7,

$$\left[A^{-1}\mathbf{1}\right]_i = 2H_{n-1} + n\bar{\rho},$$

and therefore,

$$\left[A^{-1}\mathbf{1}\right]_i \frac{\mathbf{1}^\top A^{-1}\,\mathbf{z}(S)}{\mathbf{1}^\top A^{-1}\mathbf{1}} = \frac{2|S|H_{n-1}}{n} + \frac{|S|n\bar{\rho}}{n} = \frac{2|S|H_{n-1}}{n} + |S|\bar{\rho}. \quad \square$$

Proposition 9. *For all $S \subseteq N$ with $0 < |S| < n$ and $i \in S$,*

$$w_i^{\mathrm{UKS}}(S) = \frac{1}{\binom{n}{|S|}|S|}.$$

Proof. From Proposition 7, we obtain

$$\frac{w_i^{\mathrm{UKS}}(S)}{p^{\mathrm{UKS}}(S)} = \left[A^{-1}\,\mathbf{z}(S)\right]_i - \left[A^{-1}\mathbf{1}\right]_i \frac{\mathbf{1}^\top A^{-1}\,\mathbf{z}(S)}{\mathbf{1}^\top A^{-1}\mathbf{1}}. \tag{74}$$

Using Lemma 7, we derive

$$\left[A^{-1}\,\mathbf{z}(S)\right]_i = 2H_{n-1} + |S|\bar{\rho}. \tag{75}$$

Rewriting the Shapley kernel from (42) to follow our general notation for cooperative games results in

$$\omega(S) = \frac{n-1}{\binom{n}{|S|}\,|S|\,(n-|S|)}. \tag{76}$$

 doi:10.20944/preprints202601.2308.v1

Combining the results from Lemmas 6 and 8 as well as (41), (74), (75), and (76) results in

$$\begin{aligned} w_i^{\mathrm{UKS}}(S) &= p^{\mathrm{UKS}}(S)\,\frac{w_i^{\mathrm{UKS}}(S)}{p^{\mathrm{UKS}}(S)} \\ &= Q^{-1}\,\omega(S)\left(\left[A^{-1}\,\mathbf{z}(S)\right]_i - \left[A^{-1}\mathbf{1}\right]_i \frac{\mathbf{1}^\top A^{-1}\,\mathbf{z}(S)}{\mathbf{1}^\top A^{-1}\mathbf{1}}\right) \\ &= \frac{n}{2\,(n-1)\,H_{n-1}}\,\frac{n-1}{\binom{n}{|S|}\,|S|\,(n-|S|)}\left(2H_{n-1} + |S|\bar{\rho} - \frac{2|S|H_{n-1}}{n} - |S|\bar{\rho}\right) \\ &= \frac{n}{2H_{n-1}}\,\frac{1}{\binom{n}{|S|}\,|S|\,(n-|S|)}\left(2H_{n-1} - \frac{2|S|H_{n-1}}{n}\right) \\ &= \frac{n}{2H_{n-1}}\,\frac{1}{\binom{n}{|S|}\,|S|\,(n-|S|)}\,\frac{2\,(n-|S|)\,H_{n-1}}{n} = \frac{1}{\binom{n}{|S|}|S|}. \quad \square \end{aligned}$$

Proposition 10. *For all $S \subseteq N$ with $0 < |S| < n$ and $i \notin S$,*

$$w_i^{\mathrm{UKS}}(S) = -\frac{1}{\binom{n}{|S|}(n-|S|)}.$$

Proof. From Proposition 7, we have

$$\frac{w_i^{\mathrm{UKS}}(S)}{p^{\mathrm{UKS}}(S)} = \left[A^{-1}\,\mathbf{z}(S)\right]_i - \left[A^{-1}\mathbf{1}\right]_i \frac{\mathbf{1}^\top A^{-1}\,\mathbf{z}(S)}{\mathbf{1}^\top A^{-1}\mathbf{1}}. \tag{77}$$

Using Lemma 7, we derive

$$\left[A^{-1}\,\mathbf{z}(S)\right]_i = |S|\bar{\rho}. \tag{78}$$

Combining the results from Lemmas 6 and 8 as well as (41), (76), (77), and (78) results in

$$\begin{aligned} w_i^{\mathrm{UKS}}(S) &= p^{\mathrm{UKS}}(S)\,\frac{w_i^{\mathrm{UKS}}(S)}{p^{\mathrm{UKS}}(S)} \\ &= Q^{-1}\,\omega(S)\left(\left[A^{-1}\,\mathbf{z}(S)\right]_i - \left[A^{-1}\mathbf{1}\right]_i \frac{\mathbf{1}^\top A^{-1} z(S)}{\mathbf{1}^\top A^{-1}\mathbf{1}}\right) \\ &= \frac{n}{2\,(n-1)\,H_{n-1}}\,\frac{n-1}{\binom{n}{|S|}\,|S|\,(n-|S|)}\left(|S|\bar{\rho} - \frac{2|S|H_{n-1}}{n} - |S|\bar{\rho}\right) \\ &= -\frac{n}{2H_{n-1}}\,\frac{1}{\binom{n}{|S|}\,|S|\,(n-|S|)}\,\frac{2|S|H_{n-1}}{n} = -\frac{1}{\binom{n}{|S|}(n-|S|)}. \quad \square \end{aligned}$$

Theorem 4. *The estimators obtained via LSS and UKS are both importance sampling estimators for the same linear solution concept in a sense of Theorem 1, differing only in their respective sampling strategies.*

Proof. Per definition given in (30), LSS has the form

$$\hat{\phi}_i = \frac{v(N)}{n} + \frac{1}{\tau}\sum_{\mathcal{S}\in\mathbb{S}} \frac{w_i(\mathcal{S})}{p(\mathcal{S})}\,v(\mathcal{S}),$$

with weights given by (68).

Proposition 7 shows that UKS can also be written in this form, sharing the same constant term $t = \frac{v(N)}{n}$, see Proposition 8. Moreover, by combining propositions 9 and 10, one obtains

$$w_i^{\mathrm{UKS}}(S) = \begin{cases} \frac{1}{|S|\binom{n}{|S|}} & \text{if } i \in S \\ -\frac{1}{(n-|S|)\binom{n}{|S|}} & \text{if } i \notin S \end{cases} \tag{79}$$

 doi:10.20944/preprints202601.2308.v1

for $0 < |S| < n$, which is identical to the weight function of LSS defined in (68).

Therefore, we conclude that both algorithms are importance sampling estimators of the same problem

$$\phi_i = \frac{v(N)}{n} + \sum_{\substack{S \subseteq N \\ 0<|S|<n}} w_i(S)\, v(S)$$

with $w_i(S) = w_i^{\text{LSS}}(S) = w_i^{\text{UKS}}(S)$.

Finally, we highlight that both algorithms differ only in their respective sampling distributions, since (28) and the distribution that is proportional to (76) are clearly different sampling distributions. □

We illustrate the result from Theorem 4 in Figure 3 which visualizes the different sampling strategies of LSS and UKS for $n = 9$.

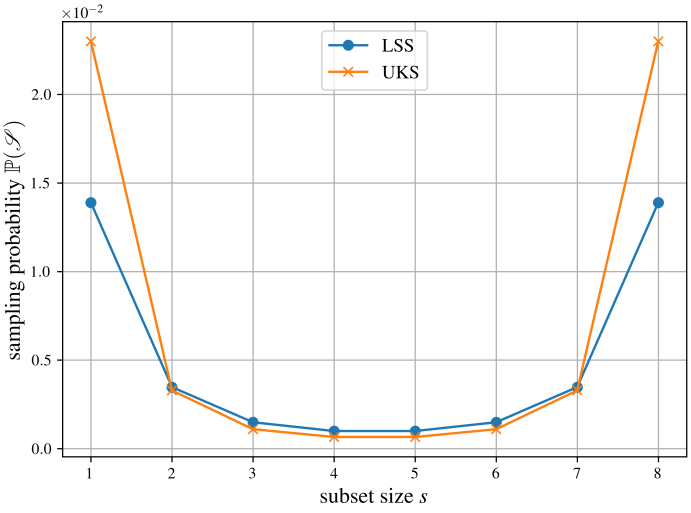

Figure 3. The different sampling distributions of LSS and UKS for $n = 9$.

Variance Comparison

In the previous section, we showed that UKS fits the importance sampling framework introduced by Theorem 1, see Proposition 7 and Theorem 4. Therefore, although Covert and Lee [7] already provided (49) as the variance for the UKS estimator, we can now use a different formulation:

Proposition 11. *The variance of the UKS estimator is given by*

$$\text{Var}[\hat{\phi}_i] = \frac{1}{\tau}\left(\sum_{\substack{S \subseteq N \\ 0<|S|<n}} \frac{w_i(S)^2}{p(S)}\, v(S)^2 - \left(\phi_i - \frac{v(N)}{n}\right)^2 \right) \quad (\forall i \in N)$$

with

$$\frac{w_i(S)^2}{p(S)} = \begin{cases} \frac{2\,(n-|S|)\,H_{n-1}}{n|S|\binom{n}{|S|}} & \text{if } i \in S \\ \frac{2|S|H_{n-1}}{n\,(n-|S|)\binom{n}{|S|}} & \text{if } i \notin S. \end{cases} \tag{80}$$

 doi:10.20944/preprints202601.2308.v1

Proof. First, let us analyze the case $i \in S$. Using Lemma 8 as well as (69), (75), and (79), we derive

$$\begin{aligned}\frac{w(S)^2}{p(S)} &= w(S)\,\frac{w(S)}{p(S)} \\ &= \frac{1}{|S|\binom{n}{|S|}}\left(\left[A^{-1}\,\mathbf{z}(S)\right]_i - \left[A^{-1}\mathbf{1}\right]_i \frac{\mathbf{1}^\top A^{-1}\,\mathbf{z}(S)}{\mathbf{1}^\top A^{-1}\mathbf{1}}\right) \\ &= \frac{1}{|S|\binom{n}{|S|}}\left(2H_{n-1} + |S|\overline{\rho} - \frac{2|S|H_{n-1}}{n} - |S|\overline{\rho}\right) \\ &= \frac{1}{|S|\binom{n}{|S|}}\left(2H_{n-1} - \frac{2|S|H_{n-1}}{n}\right) = \frac{2\,(n-|S|)\,H_{n-1}}{n|S|\binom{n}{|S|}}.\end{aligned}$$

Similarly, for the case $i \notin S$, by using Lemma 8 as well as (69), (78), and (79), we obtain

$$\begin{aligned}\frac{w(S)^2}{p(S)} &= w(S)\,\frac{w(S)}{p(S)} \\ &= -\frac{1}{(n-|S|)\binom{n}{|S|}}\left(\left[A^{-1}\,\mathbf{z}(S)\right]_i - \left[A^{-1}\mathbf{1}\right]_i \frac{\mathbf{1}^\top A^{-1}\,\mathbf{z}(S)}{\mathbf{1}^\top A^{-1}\mathbf{1}}\right) \\ &= -\frac{1}{(n-|S|)\binom{n}{|S|}}\left(|S|\overline{\rho} - \frac{2|S|H_{n-1}}{n} - |S|\overline{\rho}\right) = \frac{2|S|H_{n-1}}{n\,(n-|S|)\binom{n}{|S|}}.\end{aligned}$$

Combining both cases results in (80), and since UKS fits the framework introduced by Theorem 1, we can use (18) to compute its variance. Note that we construct a temporary solution concept $\phi_i - \frac{v(N)}{n}$ that excludes $\frac{v(N)}{n}$, similar to the variance of LSS given by Proposition 2. □

In Figure 4, we use Proposition 11 to compute the variances of UKS and Proposition 2 for those of LSS. As expected, neither LSS nor UKS outperforms the other, since they implement the same algorithm while differing only in their respective sampling strategies. The effectiveness of any given sampling strategy depends on the characteristics of the underlying problem, such as the parameters of the weighted voting game. Obviously, there exist problems for which sampling strategies other than the one employed in LSS or UKS achieve superior performance. More broadly, this illustrates that no single sampling strategy is universally optimal, and selecting the most effective one requires tailoring it to the specific problem.

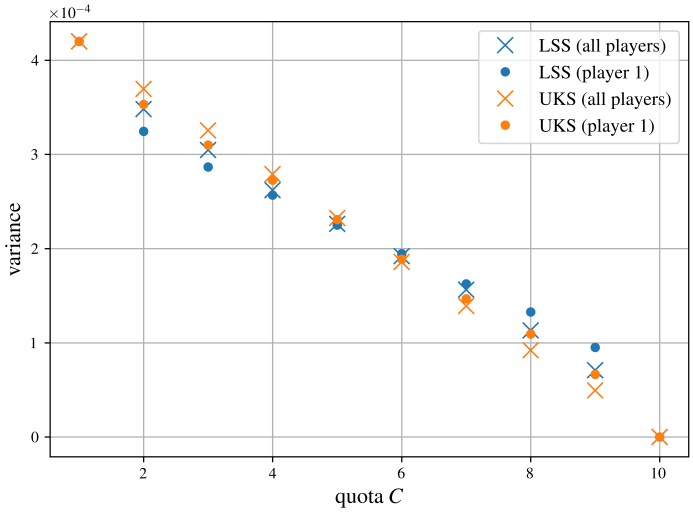

Figure 4. Theoretical variance comparison of LSS and UKS evaluated the weighted voting games defined by (7). The number of evaluations of v is $\tau = 10000$. The crosses represent the mean variance across all players' Shapley values, while the dots denote the variance for player $i = 1$.

 doi:10.20944/preprints202601.2308.v1

6. Empirical Results

We validate our results by applying the algorithms introduced in Subsection 5.2 to approximate Shapley values for airport games, weighted voting games, and three real-world interpretable machine learning problems.

The algorithms introduced in Subsection 5.2 were implemented in Python. Both the implementations and test problems can be freely accessed via the first author's GitHub repository

https://github.com/tim-pollmann/shapley-least-squares (accessed on 27 January 2026).

We consistently assume that all players' Shapley values must be estimated. Although not universal, this is common in explainable machine learning, where one seeks to explain a prediction using all features.

Before proceeding, we note that each algorithm uses different parameters to determine the number of sampled coalitions and thus, evaluations of the characteristic function v. For fair comparisons, we introduce a unified sample budget T, ensuring each algorithm performs T evaluations of v, up to negligible rounding errors.

The overall sample budget T, introduced above, sets the total number of v evaluations per algorithm. We now express the parameters of each algorithm from Subsection 5.2 in terms of T.

For LSS (Algorithm 1) and UKS (see the end of Subsection 5.2), this is a one-to-one mapping, i.e., $\tau = T$. Note that we ignore the single evaluation of v needed for calculating the initial term $\frac{v(N)}{n}$, since it is negligible in comparison to the rounding errors of other algorithms, especially for large T.

S-LSS (Algorithm 2) divides the sample space based on $n-1$ distinct values of s, for each $i \in N$. We use the heuristically motivated proportional sample allocation from (58) to obtain

$$\tau_{i,s} = \left\lceil \frac{2sT}{n^2(n-1)} \right\rceil \quad (\forall i \in N, \forall s \in \{1, \ldots, n-1\}).$$

Finally, SRS-LSS (Algorithm 3) divides the sample space into $n-1$ distinct sets. Thus, for simplicity, noting that alternative sample allocation schemes across strata may provide better results, we set

$$\tau_s = \left\lceil \frac{T}{n-1} \right\rceil \quad (\forall s \in \{1, \ldots, n-1\}).$$

We conclude that the deviation from the true total sample budget T is 1 for LSS and UKS, while the upper bounds of their deviations are given by $n^2 - n - 1$ for S-LSS and $n-2$ for SRS-LSS. We consider these deviations to be negligible in our subsequent analysis, in particular for large T.

Note that for SRS-LSS (Algorithm 3), it is not guaranteed that the algorithm runs successfully in the sense that every stratum receives at least one sample, compare Proposition 4. In our mean squared error comparisons, we mandate for any τ that at least half of all runs must be successful for the results to be displayed in the final figure.

With these definitions established, we proceeded with our experiments.

Figure 5 confirms the theoretical variances of LSS and S-LSS presented in Propositions 2 and 3, respectively. Moreover, it underscores our results from Theorem 2, where we demonstrated that one or the other algorithm might achieve a smaller variance, depending on the underlying cooperative game. Furthermore, we validate that LSS and UKS perform as expected with respect to the theoretical variances presented in Propositions 2 and 11 as Figure 5 clearly illustrates that the derived theoretical variances are accurate and that, depending on the specific problem, either LSS or UKS may slightly outperform the other, as one sampling strategy might be better suited to the given task than the other method, see Theorem 4.

 doi:10.20944/preprints202601.2308.v1

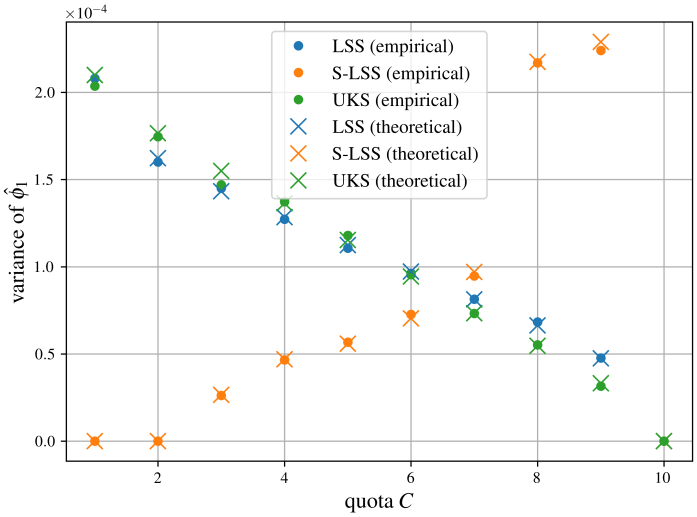

Figure 5. Empirical variance validation of $\hat{\phi}_1$ obtained via LSS, S-LSS and UKS evaluated on the weighted voting games defined by (7). The overall sample budget is $T = 10000$. The crosses represent the theoretical variances, while the dots denote the empirical variances. The empirical variances were obtained over 5000 runs.

As for mean squared error comparisons, we first look at an airport game with 100 players specified in Castro et al. [31] and compare the approximation methods from Subsection 5.2 in Figure 6. Note that the graph for SRS-LSS begins only at a sample size of 50,000, as Proposition 4 fails too frequently with smaller sample budgets. With $n = 100$ players, guaranteeing at least one sample per stratum becomes more challenging.

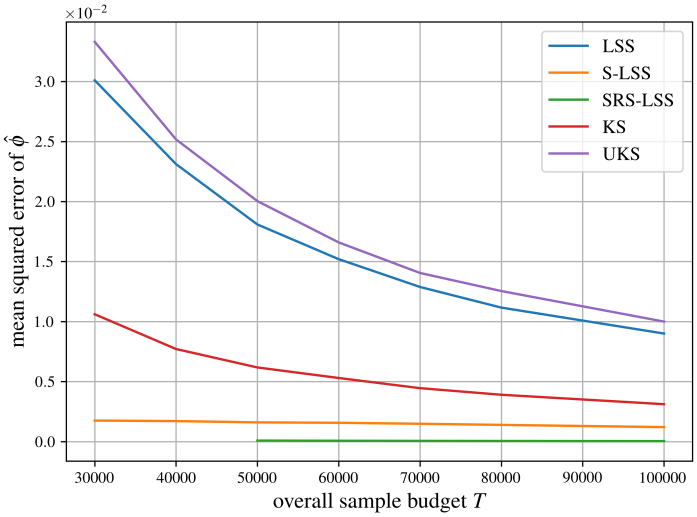

Figure 6. Empirical mean squared error comparison of $\hat{\boldsymbol{\phi}}$ obtained via LSS, S-LSS, SRS-LSS, KS and UKS evaluated on an airport game with 100 players. The mean squared errors were averaged over 250 runs.

Figure 7 compares mean squared errors for our Monte Carlo estimators from Subsection 5.2 for a weighted voting game with 50 players and normally distributed weights, which was previously employed in the software EPIC [22,23]. It can be found on the GitHub page of the second author via https://github.com/jhstaudacher/EPIC/blob/master/test_cases/normal_sqrd/normal_sqrd.n50.q249646.csv.

 doi:10.20944/preprints202601.2308.v1

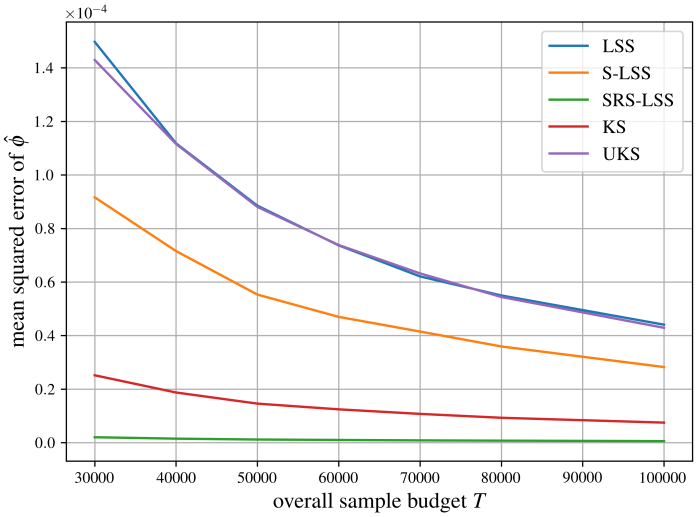

Figure 7. Empirical mean squared error comparison of $\hat{\boldsymbol{\phi}}$ obtained via LSS, S-LSS, SRS-LSS, KS and UKS evaluated on a weighted voting game with 50 players. The mean squared errors were averaged over 250 runs.

Figure 8 compares mean squared errors for our Monte Carlo estimators from Subsection 5.2 for a weighted voting game with 150 players and uniformly distributed weights, which was previously employed in the software EPIC [22,23]. It can be found on the GitHub page of the second author via https://github.com/jhstaudacher/EPIC/blob/master/test_cases/uniform/uniform.n150.q35951.csv. Note that the graph for SRS-LSS begins only at a sample size of 150,000, as Proposition 4 fails too frequently with smaller sample budgets. With $n = 150$ players, guaranteeing at least one sample per stratum becomes even more challenging than in the airport game from Figure 6.

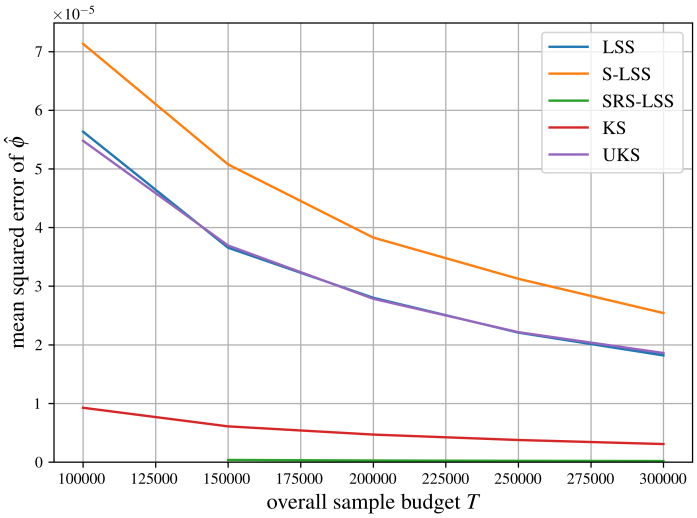

Figure 8. Empirical mean squared error comparison of $\hat{\boldsymbol{\phi}}$ obtained via LSS, S-LSS, SRS-LSS, KS and UKS evaluated on a weighted voting game with 150 players. The mean squared errors were averaged over 250 runs.

Figure 9 uses the standard *diabetes dataset* (442 patients, 10 baseline features) where the target is disease progression after one year. We train a Gradient Boosting Regressor and evaluate the approximation methods from Subsection 5.2 by their mean squared error against exact reference Shapley values.

In Figure 10, the *California housing dataset* (20,640 entries, 8 features) predicts median house value in hundreds of thousands of dollars. Using an MLP Regressor, we compare approximation methods by their mean squared error on Shapley value estimation against exact references.

 doi:10.20944/preprints202601.2308.v1

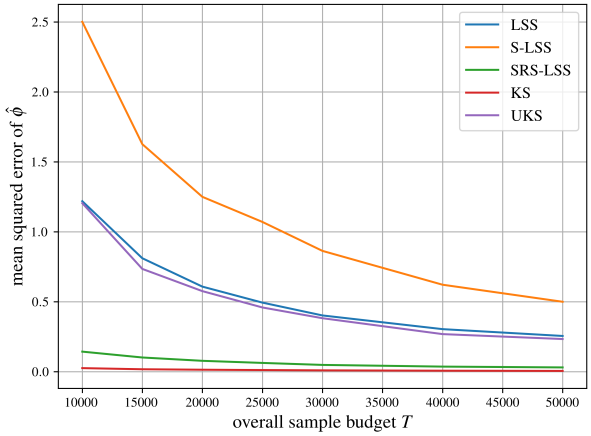

Figure 9. Empirical mean squared error comparison of $\hat{\phi}$ obtained via LSS, S-LSS, SRS-LSS, KS and UKS evaluated on the diabetes example. The mean squared errors were averaged over 250 runs.

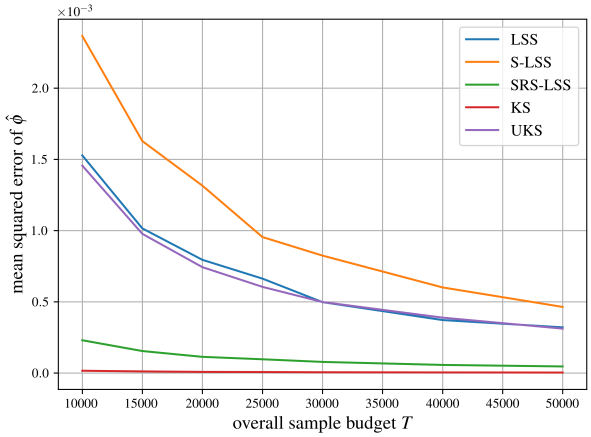

Figure 10. Empirical mean squared error comparison of $\hat{\phi}$ obtained via LSS, S-LSS, SRS-LSS, KS and UKS evaluated on the California housing example. The mean squared errors were averaged over 250 runs.

Figure 11 uses the classic *wine dataset* (178 instances, 13 features, 3 wine classes). We train a Random Forest Classifier and evaluate the approximation methods from Subsection 5.2 via mean squared error in estimating Shapley values for predicting the *probability of class 0 only*.

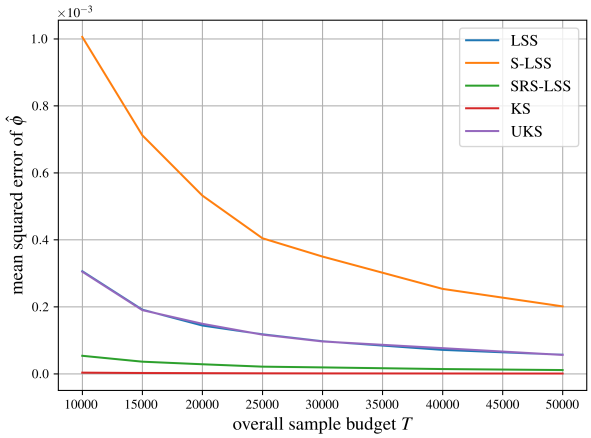

Figure 11. Empirical mean squared error comparison of $\hat{\phi}$ obtained via LSS, S-LSS, SRS-LSS, KS and UKS evaluated on the wine example. The mean squared errors were averaged over 250 runs.

 doi:10.20944/preprints202601.2308.v1

Let us summarize the comparisons of mean squared errors of all algorithms for approximating the Shapley value from Subsection 5.2 in Figures 6, 7, 8, 9, 10 and 11 succinctly. As expected, LSS and UKS perform more or less equally for all six test problems. SRS-LSS outperforms the other three provably unbiased algorithms LSS, S-LSS and UKS for all test problems. This observation is consistent with the results shown in figure 1 in Benati et al. [14], where only LSS and SRS-LSS were compared. Therefore, we conclude that the covariance terms established in Theorem 3 do not significantly affect the overall variances of individual players' Shapley value estimators negatively. S-LSS performs worst for five test problems with the notable exception of the airport game with 100 players in Figure 6 where S-LSS is even faster than KernelSHAP (KS). Not surprisingly, KS is consistently faster than UKS and LSS. For the three large cooperative games in Figures 6, 7 and 8 SRS-LSS converges faster than KS, whereas that comparison reverses for the machine learning tasks in Figures 9, 10 and 11.

7. Summary, Conclusions and Outlook

The celebrated KernelSHAP approach by Lundberg and Lee [1] as well as its later refinement Unbiased KernelSHAP (UKS) proposed by Covert and Lee [7] compute the Shapley value as a least squares optimization problem. While these two algorithms are extremely well established in the machine learning community, the methods from the paper by Benati et al. [14] — which are also based on the least square formula for the Shapley value — have received fairly little attention. To our knowledge, we are the first to apply them in the context of explainable artificial intelligence. We formulate the ideas from Benati et al. [14] as approximation algorithms for general TU games. We refer to their weighted sampling strategy as LSS (Least Square Sampling). As for their approach towards stratification, we distinguish S-LSS with no reuse of samples across strata and SRS-LSS with sample reuse across strata as proposed in Benati et al. [14].

As the result of our thorough and detailed analysis of LSS, S-LSS and SRS-LSS, we presented three key findings.

First, in Proposition 5, we showed that S-LSS as proposed in [14] is not a valid stratified variant of LSS in the sense of the definition provided Subsection 3.2,i.e., for S-LSS the strata overlap. Therefore, we demonstrated that S-LSS might reduce the variance of the obtained estimator, but it could also lead to an increase in comparison to LSS, see Theorem 2.

Second, in Theorem 3, we showed that the SRS-LSS approach proposed by Benati et al. [14] introduces covariance terms between stratum estimators, making its theoretical variance difficult to analyze. Although empirical results suggest that the variance is significantly reduced in comparison to LSS and S-LSS, a theoretical analysis remains an open research question.

Third, in Theorem 4, we established that LSS and UKS are importance sampling estimators in the sense of Theorem 1, addressing the same underlying problem but differing in their respective sampling strategies. Therefore, neither LSS nor UKS is superior over the other one. Which algorithm's variance is smaller depends on the underlying problem and whether the respective sampling strategy is suitable for this problem or not.

As noted in the introduction, this paper's aim is neither to propose a new Shapley value estimator nor to compare numerous approximation algorithms. Rather, the focus is on providing structural, theoretical, and algorithmic insight into the methods from Benati et al. [14]. While antithetic sampling — as discussed, for instance, in [32] — could be integrated into the SRS-LSS algorithm, this would have vastly exceeded the scope of this study. In the future, it could be worthwhile to investigate whether there are algorithmic approaches comparable to sophisticated stratification strategies [28,33,34] which could be successfully incorporated to enhance the performance of SRS-LSS. Definitely, our theoretical and numerical results suggest that the SRS-LSS estimator — which is unbiased — justifies more attention. Another, perhaps even more pressing question, is why SRS-LSS outperforms KernelSHAP for weighted voting games and airport games while KernelSHAP exhibits superior performance for our test problems from interpretable machine learning. We have yet to identify the properties of the characteristic function responsible for this phenomenon.

 doi:10.20944/preprints202601.2308.v1

Author Contributions: Conceptualization, T.P. and J.S.; Methodology, T.P. and J.S.; Software, T.P.; Validation, J.S.; Formal Analysis, T.P. and J.S.; Writing—original draft, T.P. and J.S.; Writing—review & editing,, T.P. and J.S.. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

Symbols

The following symbols are used in this manuscript:

$\mathcal{U}$	uniform distribution
$\odot$	Hadamard product of vectors
$\hat{\cdot}$	estimator or estimated value
$\mathbf{1}$	all-ones vector with dimension being clear from context
$\mathbf{0}$	all-zeros vector with dimension being clear from context
$\mathbb{1}_{\text{condition}}$	indicator function, 1 if *condition* is satisfied, 0 otherwise
n	number of players
N	player set, $N = \{1, \ldots, n\}$
v	characteristic function, $v : 2^N \to \mathbb{R}$
S	coalition, $S \subseteq N$
$\mathbf{z}(S)$	indicator vector corresponding to coalition S, $\mathbf{z}(S) = [\mathbb{1}_{i \in S}]_{i=1}^n$
$S(\mathbf{z})$	subset corresponding to indicator vector $\mathbf{z}$, $S(\mathbf{z}) = \{i \in N \mid z_i = 1\}$
$\boldsymbol{w}$	weight vector, $\boldsymbol{w}(S) = [w_1(S), \ldots, w_n(S)]^\top$
$\boldsymbol{\nu}$	value vector, $\boldsymbol{\nu}(S, v) = [\nu_1(S, v), \ldots, \nu_n(S, v)]^\top$
$\boldsymbol{\alpha}$	linear solution concept, $\boldsymbol{\alpha}(N, v) = \sum_{S \subseteq N} \boldsymbol{w}(S) \odot \boldsymbol{\nu}(S, v)$
$\boldsymbol{\phi}$	Shapley value
p	probability mass function used for sample generation, $p : 2^N \to [0, 1]$
τ	sample size
$\mathcal{S}$	sampled coalition, $\mathcal{S} \subseteq N$
$\mathbb{S}$	sample consisting of τ coalitions, $\mathbb{S} \sim_{\text{iid}} p$, $\mathbb{S} = [\mathcal{S}_1, \ldots, \mathcal{S}_\tau]$

Appendix A. Comparison of Our Results Against the SHAPley Interaction Quantification (SHAP-IQ) Formulation of UKS

Fumagalli et al. [30] state that choosing the probability mass function

$$p'(S) = \begin{cases} \frac{1}{2H_{n-1}} \frac{1}{n-1} \binom{n-2}{|S|-1}^{-1} & \text{if } 0 < |S| < n \\ 0 & \text{else} \end{cases}$$

with H_{n-1} being the $(n-1)$-th harmonic number ensures that UKS is equal *SHAP-IQ* for the Shapley value.

In detail, they state that the estimator obtained via UKS can be rewritten as

$$\hat{\phi}_i = \frac{v(N)}{n} + \frac{2H_{n-1}}{\tau} \sum_{\mathcal{S} \in \mathbb{S}} \left(\mathbb{1}_{i \in \mathcal{S}} - \frac{|S|}{n} \right) v(\mathcal{S}), \tag{A1}$$

where $\mathbb{S} \sim_{\text{iid}} p'$.

 doi:10.20944/preprints202601.2308.v1

Let us now rewrite (A1) similar to (70), such that

$$\hat{\phi}_i = \frac{v(N)}{n} + \frac{1}{\tau} \sum_{\mathcal{S} \in \mathbb{S}} \underbrace{2H_{n-1} \left(\mathbb{1}_{i \in \mathcal{S}} - \frac{|\mathcal{S}|}{n} \right)}_{w_i'(\mathcal{S})/p'(\mathcal{S})} v(\mathcal{S}).$$

By comparing this formulation to (70) and Proposition 8, one obtains that it shares the same initial term $\frac{v(N)}{n}$ with the formulation of UKS derived by us.

Next, we compare the functions w_i and w_i'. For $0 < |S| < n$, we find

$$\begin{aligned} w_i'(S) &= p'(S) \, \frac{w_i'(S)}{p'(S)} \\ &= \frac{1}{2H_{n-1}} \frac{1}{n-1} \binom{n-2}{|S|-1}^{-1} 2H_{n-1} \left(\mathbb{1}_{i \in S} - \frac{|S|}{n} \right) \\ &= \frac{1}{n-1} \frac{(|S|-1)!\,(n-|S|-1)!}{(n-2)!} \left(\mathbb{1}_{i \in S} - \frac{|S|}{n} \right). \end{aligned}$$

If $i \in S$, we obtain

$$\begin{aligned} w_i'(S) &= \frac{1}{n-1} \frac{(|S|-1)!\,(n-|S|-1)!}{(n-2)!} \left(1 - \frac{|S|}{n} \right) \\ &= \frac{(|S|-1)!\,(n-|S|-1)!}{(n-2)!} \frac{n-|S|}{n\,(n-1)} \\ &= \frac{(|S|-1)!\,(n-|S|)!}{n!} = \frac{|S|!\,(n-|S|)!}{|S|n!} = \frac{1}{|S|\binom{n}{|S|}}, \end{aligned}$$

which matches the case $i \in S$ in (79).

On the other hand, if $i \notin S$, we have

$$\begin{aligned} w_i'(S) &= -\frac{1}{n-1} \frac{(|S|-1)!\,(n-|S|-1)!}{(n-2)!} \frac{|S|}{n} \\ &= -\frac{|S|!\,(n-|S|-1)!}{n!} = -\frac{|S|!\,(n-|S|)!}{n!\,(n-|S|)} = -\frac{1}{(n-|S|)\binom{n}{|S|}}, \end{aligned}$$

which matches the case $i \notin S$ in (79).

Finally, we compare the sampling distributions p and p'. While Fumagalli et al. [30] define their sampling distribution as

$$p'(S) \propto \omega'(S) = \frac{1}{(n-1)\binom{n-2}{|S|-1}}, \tag{A2}$$

we define this distribution as $p(S) \propto \omega(S)$ with ω being specified in (76). Reformulating ω results in

$$\begin{aligned} \omega(S) &= \frac{n-1}{\binom{n}{|S|}\,|S|\,(n-|S|)} = \frac{|S|!\,(n-|S|)!\,(n-1)}{n!\,|S|\,(n-|S|)} \\ &= \frac{|S|\,(|S|-1)!\,(n-|S|)\,(n-|S|-1)!\,(n-1)}{n\,(n-1)\,(n-2)!\,|S|\,(n-|S|)} \\ &= \frac{(|S|-1)!\,(n-|S|-1)!}{n\,(n-2)!} = \frac{1}{n\binom{n-2}{|S|-1}}. \end{aligned} \tag{A3}$$

By comparing (A2) and (A3), one directly obtains that these expressions only differ by a constant factor, i.e.,

$$\omega(S) = \frac{n-1}{n}\,\omega'(S),$$

 doi:10.20944/preprints202601.2308.v1

which means that the corresponding sampling distributions p and p' are equivalent.

References

1. Lundberg, S.M.; Lee, S. A unified approach to interpreting model predictions. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 4768–4777. DOI: 10.5555/3295222.3295230
2. Shapley, L. S. A value for n-person games. *Contributions to the Theory of Games* **1953**, 28, 307–317.
3. Deng, X.; Papadimitriou, C. H. On the complexity of cooperative solution concepts. *Math. Oper. Res.* **1994**, *19*, 257–266. DOI: 10.1287/moor.19.2.257
4. Faigle, U.; Kern, W. The Shapley value for cooperative games under precedence constraints. *Int. J. Game Theory* **1992**, *21*, 249–266. DOI: 10.1007/BF01258278
5. Fernández, J.R.; Algaba, E.; Bilbao, J.M.; Jiménez, A.; Jiménez, N.; López, J.J. Generating functions for computing the Myerson value. *Ann. Oper. Res.* **2002**, *9*, 143–158. DOI: 10.1023/A:1016348001805
6. Chakravarty, S.R.; Mitra, M.; Sarkar, P. *A Course on Cooperative Game Theory*, Cambridge University Press: Cambridge, UK, 2015.
7. Covert, I.; Lee, S. Improving KernelSHAP: Practical Shapley Value Estimation Using Linear Regression. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*; Banerjee, A., Fukumizu, K., Eds.; PLMR, 2021; Volume 130, pp. 3457–3465. http://proceedings.mlr.press/v130/covert21a/covert21a.pdf (accessed on 27 January 2026).
8. Charnes, A.; Golany, B.; Keane, M.; Rousseau, J. Extremal Principle Solutions of Games in Characteristic Function Form: Core, Chebychev and Shapley Value Generalizations. In *Econometrics of Planning and Efficiency*; Dordrecht: Springer Netherlands, 1988; Volume 11, pp. 123-133. DOI: 10.1007/978-3-642-32241-9_48
9. Ruiz, L.M.; Valenciano, F..; Zarzuelo, J.M. The Family of Least Square Values for Transferable Utility Games. *Games Econ. Behav.* **1998**, *24*, 109–130. DOI: 10.1006/game.1997.0622
10. Littlechild, S.C.; Thompson, G.F. Aircraft landing fees: a game theory approach. *Bell J. Econ.* **1977**, *8*, 186–204.
11. Algaba, E.; Bilbao, J.M.; Fernández-García, J.R. The distribution of power in the European Constitution. *Eur. J. Oper. Res.* **2007**, *176*, 1752–1766. DOI: 10.1016/j.ejor.2005.12.002
12. Kóczy, L.A. Beyond Lisbon. Demographic trends and voting power in the European Union Council of Ministers. *Math. Soc. Sci.* **2012**, *63*, 152–158. DOI: doi.org/10.1016/j.mathsocsci.2011.08.005
13. Kóczy, L.A. Brexit and Power in the Council of the European Union. *Games* **2021**, *12*, 51. DOI: doi.org/10.3390/g12020051
14. Benati, S.; López-Blázquez, F.; Puerto, J. A stochastic approach to approximate values in cooperative games. *Eur. J. Oper. Res.* **2019**, *279*, 93–106. DOI: 10.1016/j.ejor.2019.05.027
15. Pollmann, T.; Staudacher, J. On Importance Sampling and Multilinear Extensions for Approximating Shapley Values with Applications to Explainable Artificial Intelligence. Preprints **2026**. DOI:10.20944/preprints202601.0530.v1
16. Peters, H. Game theory: A Multi-leveled approach. 2nd ed., Springer, Berlin/Heidelberg, Germany, 2015. DOI: 10.1007/978-3-662-46950-7
17. Rozemberczki, B.; Watson, L.; Bayer, P.; Yang, H.; Kiss, O.; Nilsson, S.; Sarkar, R. The Shapley value in machine learning. In *The 31st International Joint Conference on Artificial Intelligence and the 25th European Conference on Artificial Intelligence, IJCAI-ECAI 2022*; de Raedt, L. Ed.; International Joint Conferences on Artificial Intelligence Organization; Vienna, Austria; pp. 5572–5579. DOI: 10.24963/ijcai.2022/778
18. Chen, H.; Covert, I.C.; Lundberg, S.M.; Lee, S. Algorithms to estimate Shapley value feature attributions. *Nat. Mach. Intell.* **2023**, *5*, 590–601. DOI: 10.1038/s42256-023-00657-x
19. Molnar, C.: Interpreting Machine Learning Models With SHAP. LeanPub, 2023. https://leanpub.com/shap
20. Borm, P.; Hamers, H.; Hendrickx, R. Operations research games: A survey. *Top* **2001**, *9*, 139–199. DOI:10.1007/BF02579075
21. Fatima, S.S.; Wooldridge, M.; Jennings, N.R. A linear approximation method for the Shapley value. *Artif. Intell.* **2008**, *172*, 1673–1699. DOI: 10.1016/j.artint.2008.05.003
22. Staudacher, J.; Kóczy, L.Á.; Stach, I.;Filipp, J.; Kramer, M.; Noffke, T.; Olsson, L.; Pichler, J.; Singer, T. Computing power indices for weighted voting games via dynamic programming. *Oper. Res. Dec.* **2021**, *31*, 123–145. DOI:10.37190/ord210206
23. Staudacher, J.; Wagner, F.; Filipp, J. Dynamic Programming for Computing Power Indices for Weighted Voting Games with Precoalitions. *Games* **2021**, *13*, 6. DOI:10.3390/g13010006

 doi:10.20944/preprints202601.2308.v1

24. Sundararajan, M.; Najmi, A. The Many Shapley Values for Model Explanation. In *Proceedings of the 37th International Conference on Machine Learning*; Daumé III, H., Singh, A., Eds.; PMLR: London, UK, 2020; Volume 119; pp. 9269–9278. Available online:http://proceedings.mlr.press/v119/sundararajan20b/sundararajan20b.pdf (accessed on 27 January 2026).
25. Rubinstein, R.; Kroese, D. Monte Carlo methods. *Wiley Interdiscip. Rev. Comput. Stat.* **2012**, *4*, 48–58. https://doi.org/10.1002/wics.194.
26. Botev, Z.; Ridder, A. Variance reduction. In *Wiley statsRef: Statistics Reference Online*; Wiley: Hoboken, NJ, USA, 2017; pp. 1–6. https://doi.org/10.1002/9781118445112.stat07975.
27. Rubinstein, R.Y.; Kroese, D.P. *Simulation and the Monte Carlo method*. John Wiley & Sons: Hoboken, New Jersey, USA, 2007. DOI: 10.1002/9780470230381
28. Castro, J.; Gómez, D.; Molina, E.; Tejada, J. Improving polynomial estimation of the Shapley value by stratified random sampling with optimum allocation. *Comput. Oper. Res.* **2017**, *82*, 180–188. DOI: 10.1016/j.cor.2017.01.019
29. Hunter, D. An Upper Bound for the Probability of a Union. *J. Appl. Probab.*, **1976**, *13*, 597–603. DOI: 10.2307/3212481
30. Fumagalli, F.; Muschalik, M.; Kolpaczki, P.; Hüllermeier, E.; Hammer, B. HAP-IQ: Unified Approximation of any-order Shapley Interactions. In *Advances in Neural Information Processing Systems*; Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., Levine, S., Eds.; Curran Associates, Inc.: USA, 2023; Volume 36, pp. 11515–11551.
31. Castro, J.; Gómez, D.; Tejada, J. Polynomial calculation of the Shapley value based on sampling. *Comput. Oper. Res.* **2009**, *36*, 1726–1730. DOI: 10.1016/j.cor.2008.04.004
32. Staudacher, J.; Pollmann, T. Assessing Antithetic Sampling for Approximating Shapley, Banzhaf, and Owen Values. *AppliedMath* **2023**, *3*, 957-988. DOI: 10.3390/appliedmath3040049
33. Neyman, J. On the Two Different Aspects of the Representative Method: The Method of Stratified Sampling and the Method of Purposive Selection. *J. R. Stat.* **1934**, *97*, 558–625. DOI: 10.2307/2342192
34. Burgess, M.A.; Chapman, A.C. Approximating the Shapley Value Using Stratified Empirical Bernstein Sampling. In *IJCAI*, 2021, pp. 73–81.