Supplementary Materials for Improving KernelSHAP: Practical Shapley Value Estimation via Linear Regression

1 CALCULATING A EXACTLY

Recall the definition of A, which is a term in the solution to the Shapley value linear regression problem:

$$A = \mathbb{E}[ZZ^T].$$

The entries of A are straightforward to calculate because Z is a random binary vector with a known distribution. Recall that Z is distributed according to $p(Z)$, which is defined as:

$$p(z) = \begin{cases} Q^{-1}\mu_{\text{Sh}}(Z) & 0 < \mathbf{1}^T z < d \\ 0 & \text{otherwise,} \end{cases}$$

where the normalizing constant Q is given by:

$$\begin{aligned} Q &= \sum_{0<\mathbf{1}^T z<d} \mu_{\text{Sh}}(z) \\ &= \sum_{k=1}^{d-1} \binom{d}{k} \frac{d-1}{\binom{d}{k} k(d-k)} \\ &= (d-1) \sum_{k=1}^{d-1} \frac{1}{k(d-k)}. \end{aligned}$$

Although Q does not have a simple closed-form solution, the expression above can be calculated numerically. The diagonal entries A_{ii} are then given by:

$$\begin{aligned} A_{ii} &= \mathbb{E}[Z_i Z_i] = p(Z_i = 1) \\ &= \sum_{k=1}^{d-1} p(Z_i = 1 | \mathbf{1}^T Z = k) p(\mathbf{1}^T Z = k) \\ &= \sum_{k=1}^{d-1} \frac{\binom{d-1}{k-1}}{\binom{d}{k}} \cdot Q^{-1} \binom{d}{k} \frac{d-1}{\binom{d}{k} k(d-k)} \\ &= \frac{\sum_{k=1}^{d-1} \frac{1}{d(d-k)}}{\sum_{k=1}^{d-1} \frac{1}{k(d-k)}}. \end{aligned}$$

This is equal to $\frac{1}{2}$ regardless of the value of d. To see this, consider the probability $p(Z_i = 0)$:

$$
\begin{aligned}
p(Z_i = 0) &= 1 - p(Z_i = 1) \\
&= 1 - \frac{\sum_{k=1}^{d-1} \frac{1}{d(d-k)}}{\sum_{k=1}^{d-1} \frac{1}{k(d-k)}} \\
&= \frac{\sum_{k=1}^{d-1} \frac{1}{d(d-k)}}{\sum_{k=1}^{d-1} \frac{1}{k(d-k)}} \\
&= p(Z_i = 1) \\
\Rightarrow A_{ii} &= \frac{1}{2}.
\end{aligned}
$$

Next, consider the off-diagonal entries A_{ij} for $i \neq j$:

$$
\begin{aligned}
A_{ij} &= \mathbb{E}[Z_i Z_j] = p(Z_i = Z_j = 1) \\
&= \sum_{k=2}^{d-1} p(Z_i = Z_j = 1 | \mathbf{1}^T Z = k) p(\mathbf{1}^T Z = k) \\
&= \sum_{k=2}^{d-1} \frac{\binom{d-2}{k-2}}{\binom{d}{k}} \cdot Q^{-1} \binom{d}{k} \frac{d-1}{\binom{d}{k} k(d-k)} \\
&= \frac{1}{d(d-1)} \frac{\sum_{k=2}^{d-1} \frac{k-1}{d-k}}{\sum_{k=1}^{d-1} \frac{1}{k(d-k)}}.
\end{aligned}
$$

The value for off-diagonal entries A_{ij} depends on d, unlike the diagonal entries A_{ii}. Although it does not have a simple closed-form expression, this value can be calculated numerically in $\mathcal{O}(d)$ time.

2 VARIANCE REDUCTION PROOF

We present a proof for Theorem 1, and we prove that a weaker condition than $G_v \succeq 0$ holds for all cooperative games (the diagonal elements satisfy $(G_v)_{ii} \geq 0$ for all games v).

2.1 Theorem 1 Proof

In Section 4.2, we proposed a variance reduction technique that pairs each sample $z_i \sim p(Z)$ with its complement $\mathbf{1} - z_i$ when estimating b. We now provide a proof for the condition that must be satisfied for the estimator $\check{\beta}_n$ to have lower variance than $\bar{\beta}_n$. As mentioned in the main text, the multivariate CLT asserts that

$$
\begin{aligned}
\bar{b}_n \sqrt{n} &\xrightarrow{D} \mathcal{N}(b, \Sigma_{\bar{b}}) \\
\check{b}_n \sqrt{n} &\xrightarrow{D} \mathcal{N}(b, \Sigma_{\check{b}}),
\end{aligned}
$$

where

$$
\begin{aligned}
\Sigma_{\bar{b}} &= \mathrm{Cov}\big(Z v(Z)\big), \\
\Sigma_{\check{b}} &= \mathrm{Cov}\Big(\frac{1}{2}\big(Z v(Z) + (\mathbf{1} - Z) v(\mathbf{1} - Z)\big)\Big).
\end{aligned}
$$

We can also apply the multivariate CLT to the Shapley value estimators $\bar{\beta}_n$ and $\check{\beta}_n$. We can see that

$$\bar{\beta}_n\sqrt{n} \xrightarrow{D} \mathcal{N}(\beta^*, \Sigma_{\bar{\beta}})$$
$$\check{\beta}_n\sqrt{n} \xrightarrow{D} \mathcal{N}(\beta^*, \Sigma_{\check{\beta}}),$$

where, due to their multiplicative dependence on b estimators, the covariance matrices are defined as

$$\Sigma_{\bar{\beta}} = C\Sigma_{\bar{b}}C^T$$
$$\Sigma_{\check{\beta}} = C\Sigma_{\check{b}}C^T.$$

Next, we examine the relationship between $\Sigma_{\bar{b}}$ and $\Sigma_{\check{b}}$ because they dictate the relationship between $\Sigma_{\bar{\beta}}$ and $\Sigma_{\check{\beta}}$. To simplify our notation, we introduce three jointly distributed random variables, M^0, M^1 and $\bar{M}$, which are all functions of the random variable Z:

$$M^0 = Zv(Z) - \mathbb{E}[Z]v(\mathbf{0})$$
$$M^1 = (\mathbf{1} - Z)v(\mathbf{1} - Z) - \mathbb{E}[\mathbf{1} - Z]v(\mathbf{0})$$
$$\bar{M} = \frac{1}{2}\big(M^0 + M^1\big).$$

To understand $\bar{M}$'s covariance structure, we can decompose it using standard covariance properties and the fact that $p(z) = p(\mathbf{1} - z)$ for all z:

$$\begin{aligned}
\mathrm{Cov}(\bar{M}, \bar{M})_{ij} &= \frac{1}{4}\mathrm{Cov}(M_i^0 + M_i^1, M_j^0 + M_j^1) \\
&= \frac{1}{4}\Big(\mathrm{Cov}(M_i^0, M_j^0) + \mathrm{Cov}(M_i^1, \bar{M}_j^1) + \mathrm{Cov}(M_i^0, M_j^1) + \mathrm{Cov}(M_i^1, M_j^0)\Big) \\
&= \frac{1}{2}\Big(\mathrm{Cov}(M_i^0, M_j^0) + \mathrm{Cov}(M_i^0, M_j^1)\Big).
\end{aligned}$$

We can now compare $\Sigma_{\bar{b}}$ to $\Sigma_{\check{b}}$. To account for each $\bar{M}$ sample requiring twice as many cooperative game evaluations as M^0, we compare the covariance $\mathrm{Cov}(\bar{b}_{2n})$ to the covariance $\mathrm{Cov}(\check{b}_n)$:

$$n\Big(\mathrm{Cov}(\bar{b}_{2n}) - \mathrm{Cov}(\check{b}_n)\Big)_{ij} = -\frac{1}{2}\mathrm{Cov}(M_i^0, M_j^1).$$

Based on this, we define G_v as follows:

$$\begin{aligned}
G_v &= -\mathrm{Cov}(M_i^0, M_j^1) \\
&= -\mathrm{Cov}\Big(Zv(Z) - \mathbb{E}[Z]v(\mathbf{0}), (\mathbf{1} - Z)v(\mathbf{1} - Z) - \mathbb{E}[\mathbf{1} - Z]v(\mathbf{0})\Big) \\
&= -\mathrm{Cov}\Big(Zv(Z), (\mathbf{1} - Z)v(\mathbf{1} - Z)\Big).
\end{aligned}$$

This is the matrix referenced in Theorem 1. Notice that G_v is the negated cross-covariance between M^0 and M^1, which is the off-diagonal block in the joint covariance matrix for the concatenated random variable (M^0, M^1). This matrix is symmetric, unlike general cross-covariance matrices, and its eigen-structure determines whether our variance reduction approach is effective. In particular, if the condition $G_v \succeq 0$ is satisfied, then we have

$$\mathrm{Cov}(\bar{b}_{2n}) \succeq \mathrm{Cov}(\check{b}_n),$$

which implies that

$$\mathrm{Cov}(\bar{\beta}_{2n}) \succeq \mathrm{Cov}(\check{\beta}_n).$$

Since the inverses of two ordered matrices are also ordered, we get the result:

$$\mathrm{Cov}(\bar{\beta}_{2n})^{-1} \preceq \mathrm{Cov}(\check{\beta}_n)^{-1}.$$

This has implications for quadratic forms involving each matrix. For any vector $a \in \mathbb{R}^d$, we have the inequality

$$a^T \mathrm{Cov}(\bar{\beta}_{2n})^{-1} a \leq a^T \mathrm{Cov}(\check{\beta}_n)^{-1} a.$$

The last inequality has a geometric interpretation. It shows that the confidence ellipsoid (i.e., the confidence region, or prediction ellipsoid) for $\check{\beta}_n$ is contained by the corresponding confidence ellipsoid for $\bar{\beta}_{2n}$ since large values of n lead each estimator to converge to its asymptotically normal distribution. This is because the confidence ellipsoids are defined for $\alpha \in (0, 1)$ as

$$\begin{aligned}
\bar{E}_{2n,\alpha} &= \Big\{ a \in \mathbb{R}^d : (a - \beta^*)^T \mathrm{Cov}(\bar{\beta}_{2n})^{-1} (a - \beta^*) \leq \sqrt{\chi^2_d(\alpha)} \Big\} \\
\check{E}_{n,\alpha} &= \Big\{ a \in \mathbb{R}^d : (a - \beta^*)^T \mathrm{Cov}(\check{\beta}_n)^{-1} (a - \beta^*) \leq \sqrt{\chi^2_d(\alpha)} \Big\},
\end{aligned}$$

where $\chi^2_d(\alpha)$ denotes the inverse CDF of a Chi-squared distribution with d degrees of freedom evaluated at α. More precisely, we have $\check{E}_{n,\alpha} \subseteq \bar{E}_{2n,\alpha}$ because

$$\begin{aligned}
&(a - \beta^*)^T \mathrm{Cov}(\check{\beta}_n)^{-1} (a - \beta^*) \leq \sqrt{\chi^2_d(\alpha)} \\
\Rightarrow &(a - \beta^*)^T \mathrm{Cov}(\bar{\beta}_{2n})^{-1} (a - \beta^*) \leq \sqrt{\chi^2_d(\alpha)}.
\end{aligned}$$

This completes the proof.

2.2 A Weaker Condition

Consider the matrix G_v, which for a game v is defined as

$$G_v = -\mathrm{Cov}\Big(Zv(Z), (\mathbf{1} - Z) v(\mathbf{1} - Z) \Big).$$

A necessary (but not sufficient) condition for $G_v \succeq 0$ is that its diagonal elements are non-negative. We can prove that this weaker condition holds for all games. For an arbitrary game v, the diagonal value $(G_v)_{ii}$ is given by:

$$
\begin{aligned}
(G_v)_{ii} &= -\mathrm{Cov}\Big(Z_i v(Z), (1-Z_i)v(\mathbf{1}-Z)\Big) \\
&= -\mathbb{E}\big[Z_i(1-Z_i)v(Z)v(\mathbf{1}-Z)\big] + \mathbb{E}\big[Z_i v(Z)\big]\mathbb{E}\big[(1-Z_i)v(\mathbf{1}-Z)\big] \\
&= \mathbb{E}\big[Z_i v(Z)\big]^2 \\
&= \mathbb{E}\big[v(S)|i \in S\big]^2 \\
&\geq 0.
\end{aligned}
$$

Geometrically, this condition means that the confidence ellipsoid $\bar{E}_{2n,\alpha}$ extends beyond the ellipsoid $\check{E}_{n,\alpha}$ in the axis-aligned directions. In a probabilistic sense, it means that the variance for each Shapley value estimate is lower when using the paired sampling technique.

3 SHAPLEY EFFECTS

Shapley Effects is a model explanation method that summarizes the model f's sensitivity to each feature [3]. It is based on the cooperative game

$$
\tilde{w}(S) = \mathrm{Var}\big(\mathbb{E}[f(X)|X_S]\big). \tag{1}
$$

To show that Shapley Effects can be viewed as the expectation of a stochastic cooperative game, we reformulate this game (Covert et al. [1]) as:

$$
\begin{aligned}
\tilde{w}(S) &= \mathrm{Var}\big(\mathbb{E}[f(X)|X_S]\big) \\
&= \mathrm{Var}\big(f(X)\big) - \mathbb{E}_{X_S}\big[\mathrm{Var}(f(X)|X_S)\big] \\
&= c - \mathbb{E}_{X_S}\Big[\mathbb{E}_{X_{D\setminus S}|X_S}\big[\big(\mathbb{E}[f(X)|X_S] - f(X_S, X_{D\setminus S})\big)^2\big]\Big] \\
&= c - \mathbb{E}_X\Big[\big(\mathbb{E}[f(X)|X_S] - f(X)\big)^2\Big].
\end{aligned}
$$

If we generalize this cooperative game to allow arbitrary loss functions (e.g., cross entropy loss for classification tasks) rather than MSE, then we can ignore the constant value and re-write the game as

$$
\tilde{w}(S) = -\mathbb{E}_X\Big[\ell\big(\mathbb{E}[f(X)|X_S], f(X)\big)\Big].
$$

Now, it is apparent that Shapley Effects is based on a cooperative game that is the expectation of a stochastic cooperative game, or $\tilde{w}(S) = \mathbb{E}_X[\tilde{W}(S,X)]$, where $\tilde{W}(S,X)$ is defined as:

$$
\tilde{W}(S,X) = -\ell\big(\mathbb{E}[f(X)|X_S], f(X)\big).
$$

Unlike the stochastic cooperative game implicitly used by SAGE, the exogenous random variable for this game is $U = X$.

4 STOCHASTIC COOPERATIVE GAME PROOFS

For a stochastic cooperative game $V(S,U)$, the generalized Shapley values are given by the expression

$$\begin{aligned}\phi_i(V) &= \frac{1}{d}\sum_{S\subseteq D\setminus\{i\}}\binom{d-1}{|S|}^{-1}\mathbb{E}_U\big[V(S\cup\{i\},U)-V(S,U)\big]\\ &= \frac{1}{d}\sum_{S\subseteq D\setminus\{i\}}\binom{d-1}{|S|}^{-1}\mathbb{E}_U\big[V(S\cup\{i\},U)\big]-\mathbb{E}_U\big[V(S,U)\big].\end{aligned}$$

The second line above shows that the generalized Shapley values are equivalent to the Shapley values of the game's expectation, or $\phi_i(\bar{V})$, where $\bar{V}(S) = \mathbb{E}_U[V(S,U)]$. Based on this, we can also understand the values $\phi_1(V),\ldots,\phi_d(V)$ as the optimal coefficients for the following weighted least squares problem:

$$\begin{aligned}&\min_{\beta_0,\ldots,\beta_d}\ \sum_z p(z)\Big(\beta_0 + z^T\beta - \mathbb{E}_U\big[V(z,U)\big]\Big)^2\\ &\text{s.t.}\quad \beta_0 = \mathbb{E}_U\big[V(\mathbf{0},U)\big],\quad \mathbf{1}^T\beta = \mathbb{E}_U\big[V(\mathbf{1},U)\big]-\mathbb{E}_U\big[V(\mathbf{0},U)\big].\end{aligned}$$

Using our derivation from the main text (Section 3.3), we can write the solution as

$$\beta^* = A^{-1}\Big(b - \mathbf{1}\frac{\mathbf{1}^TA^{-1}b - \mathbb{E}_U[V(\mathbf{1},U)] + \mathbb{E}_U[V(\mathbf{0},U)]}{\mathbf{1}^TA^{-1}\mathbf{1}}\Big),$$

where A and b are given by the expressions

$$\begin{aligned}A &= \mathbb{E}[ZZ^T]\\ b &= \mathbb{E}_Z\Big[Z\big(\mathbb{E}_U[V(Z,U)] - \mathbb{E}_U[V(\mathbf{0},U)]\big)\Big].\end{aligned}$$

Now, we consider our adaptations of KernelSHAP and unbiased KernelSHAP and examine whether these estimators are consistent or unbiased. We begin with the stochastic version of KernelSHAP presented in the main text (Section 5.3). Recall that this approach uses the original A estimator $\hat{A}_n$ and the modified b estimator $\tilde{b}_n$, which is defined as:

$$\tilde{b}_n = \frac{1}{2}\sum_{i=1}^n z_i\big(V(z_i,u_i) - \mathbb{E}_U\big[V(\mathbf{0},U)\big]\big).$$

As mentioned in the main text, the strong law of large numbers lets us conclude that $\lim_{n\to\infty}\hat{A}_n = A$. Thus, we can understand the b estimator's expectation as follows:

$$\begin{aligned}\mathbb{E}\big[\tilde{b}_n\big] &= \mathbb{E}_{ZU}\Big[Z\big(V(Z,U) - \mathbb{E}_U\big[V(\mathbf{0},U)\big]\big)\Big]\\ &= \mathbb{E}_Z\Big[Z\big(\mathbb{E}_U[V(Z,U)] - \mathbb{E}_U[V(\mathbf{0},U)]\big)\Big]\\ &= b.\end{aligned}$$

With this, we conclude that $\lim_{n\to\infty}\tilde{b}_n = b$ and that $\tilde{\beta}_n$ are consistent, or

$$\lim_{n\to\infty}\tilde{\beta}_n = \beta^*.$$

To adapt unbiased KernelSHAP to the setting of stochastic cooperative games, we use the same technique of pairing independent samples of Z and U. To estimate b, we use an estimator $\tilde{\tilde{b}}_n$ defined as:

$$\tilde{\tilde{b}}_n = \frac{1}{n}\sum_{i=1}^{n} z_i V(z_i, u_i) - \mathbb{E}\big[Z\big]\mathbb{E}_U\big[V(\mathbf{0}, U)\big].$$

We then substitute this into a Shapley value estimator as follows:

$$\tilde{\tilde{\beta}}_n = A^{-1}\Big(\tilde{\tilde{b}}_n - \mathbf{1}\frac{\mathbf{1}^T A^{-1}\tilde{\tilde{b}}_n - v(\mathbf{1}) + v(\mathbf{0})}{\mathbf{1}^T A^{-1}\mathbf{1}}\Big). \tag{2}$$

This is consistent and unbiased because of the linear dependence on $\tilde{\tilde{b}}_n$ and the fact that $\tilde{\tilde{b}}_n$ is unbiased:

$$\begin{aligned}
\mathbb{E}\big[\tilde{\tilde{b}}_n\big] &= \mathbb{E}_{ZU}\Big[ZV(Z,U) - \mathbb{E}\big[Z\big]\mathbb{E}_U\big[V(\mathbf{0},U)\big]\Big] \\
&= \mathbb{E}_Z\Big[Z\big(\mathbb{E}_U[V(Z,U)] - \mathbb{E}_U[V(\mathbf{0},U)]\big)\Big] \\
&= b.
\end{aligned}$$

With this, we conclude that $\mathbb{E}[\tilde{\tilde{\beta}}_n] = \beta^*$ and $\lim_{n\to\infty} \tilde{\tilde{\beta}}_n = \beta^*$.

5 EXPERIMENT DETAILS

Here, we provide further details about experiments described in the main body of text.

5.1 Datasets and Hyperparameters

For all three explanation methods considered in our experiments – SHAP [2], SAGE [1] and Shapley Effects [3] – we handled removed features by marginalizing them out according to their joint marginal distribution. This is the default behavior for SHAP, but it is an approximation of what is required by SAGE and Shapley Effects. However, this choice should not affect the outcome of our experiments, which focus on the convergence properties of our Shapley value estimators (and not the underlying cooperative games).

Both SAGE and Shapley Effects require a loss function (Section 3). We used the cross entropy loss for SAGE and the soft cross entropy loss for Shapley Effects.

For the breast cancer (BRCA) subtype classification dataset, we selected 100 out of 17,814 genes to avoid overfitting on the relatively small dataset size (only 510 patients). These genes were selected at random: we tried ten random seeds and selected the subset that achieved the best performance to ensure that several relevant BRCA genes were included. A small portion of missing expression values were imputed with their mean. The data was centered and normalized prior to fitting a ℓ_1 regularized logistic regression model; the regularization parameter was chosen using a validation set.

5.2 SHAP Run-time Comparison

To compare the run-time of various SHAP value estimators, we sought to compare the ratio of the mean number of samples required by each method. For a single example x whose SHAP values are represented by β^*, the mean squared estimation error can be decomposed into the variance and bias as follows:

$$\mathbb{E}\big[||\hat{\beta}_n - \beta^*||^2\big] = \mathbb{E}\big[||\hat{\beta}_n - \mathbb{E}[\hat{\beta}_n]||^2\big] + \big|\big|\mathbb{E}[\hat{\beta}_n] - \beta^*\big|\big|^2.$$

Since we found that the error is dominated by variance rather than bias (Section 4.1), we can make the following approximation to relate the error to the trace of the covariance matrix:

$$\begin{aligned}\mathbb{E}\big[||\hat{\beta}_n - \beta^*||^2\big] &= \mathbb{E}\big[||\hat{\beta}_n - \mathbb{E}[\hat{\beta}_n]||^2\big] + \big|\big|\mathbb{E}[\hat{\beta}_n] - \beta^*\big|\big|^2 \\ &\approx \mathbb{E}\big[||\hat{\beta}_n - \mathbb{E}[\hat{\beta}_n]||^2\big] \\ &= \mathrm{Tr}\Big(\mathrm{Cov}(\hat{\beta}_n)\Big). \end{aligned} \tag{3}$$

If we define convergence based on the mean estimation error falling below a threshold value t, then the convergence condition is

$$\mathbb{E}\big[||\hat{\beta}_n - \beta^*||^2\big] \leq t.$$

Using our approximation (Eq. 3), we can see that this condition is approximately equivalent to

$$\mathbb{E}\big[||\hat{\beta}_n - \beta^*||^2\big] \approx \mathrm{Tr}\Big(\mathrm{Cov}(\hat{\beta}_n)\Big) \approx \frac{\mathrm{Tr}(\Sigma_{\hat{\beta}})}{n} \leq t.$$

For a given threshold t, the mean number of samples required to explain individual predictions is therefore based on the mean trace of the covariance matrix $\Sigma_{\hat{\beta}}$ (or the analogous covariance matrix for a different estimator). To compare two methods, we simply calculate the ratio of the mean trace of the covariance matrices. These ratios are reported in Table 1, where each covariance matrix is calculated empirically across 100 runs with $n = 2048$ samples.

6 CONVERGENCE EXPERIMENTS

In Section 4.1, we empirically compared the bias and variance for the original and unbiased versions of KernelSHAP using a single census income prediction. The results (Figure 1) showed that both versions' estimation errors were dominated by variance rather than bias, and that the original version had significantly lower variance. To verify that this result is not an anomaly, we replicated it on multiple examples and across several datasets.

First, we examined several individual predictions for the census income, German credit and bank marketing datasets. To highlight the effectiveness of our paired sampling approach (Section 4.2), we added these methods as additional comparisons. Rather than decomposing the error into bias and variance as in the main text, we simply calculated the mean squared error across 100 runs of each estimator. Figure 1 shows the error for several census income predictions, Figure 3 for several bank marketing predictions, and Figure 5 for several credit quality predictions. These results confirm that the original version of KernelSHAP converges significantly faster than the unbiased version, and that the paired sampling technique is effective for both estimators. The dataset sampling approach (original KernelSHAP) appears preferable in practice despite being more difficult to analyze because it converges to the correct result much faster.

Second, we calculated a global measure of the bias and variance for each estimator using the same datasets (Table 1). Given 100 examples from each dataset, we calculated the mean bias and mean variance for each estimator empirically across 100 runs given $n = 256$ samples. Results show that the bias is nearly zero for all estimators, not just the unbiased ones; they also show that the variance is often significantly larger than the bias. However, when using the dataset sampling approach (original) in combination with the paired sampling technique, the bias and variance are comparably low (≈ 0) after 256 samples. The only exception is the unbiased estimator that does not use paired sampling, but this is likely due to estimation error because its bias is provably equal to zero.

Finally, Section 4.3 also proposed assuming that the original KernelSHAP estimator's variance reduces at a rate of $\mathcal{O}(\frac{1}{n})$, similar to the unbiased version (for which we proved this rate). Although this result is difficult to prove formally, it seems to hold empirically across multiple predictions and several datasets. In Figures 2, 4 and 6, we display the product of the estimator's variance with the number of samples for the census, bank and credit datasets. Results confirm that the product is roughly constant as the number of samples increases, indicating that the variance for all four estimators (not just the unbiased ones) reduces at a rate of $\mathcal{O}(\frac{1}{n})$.

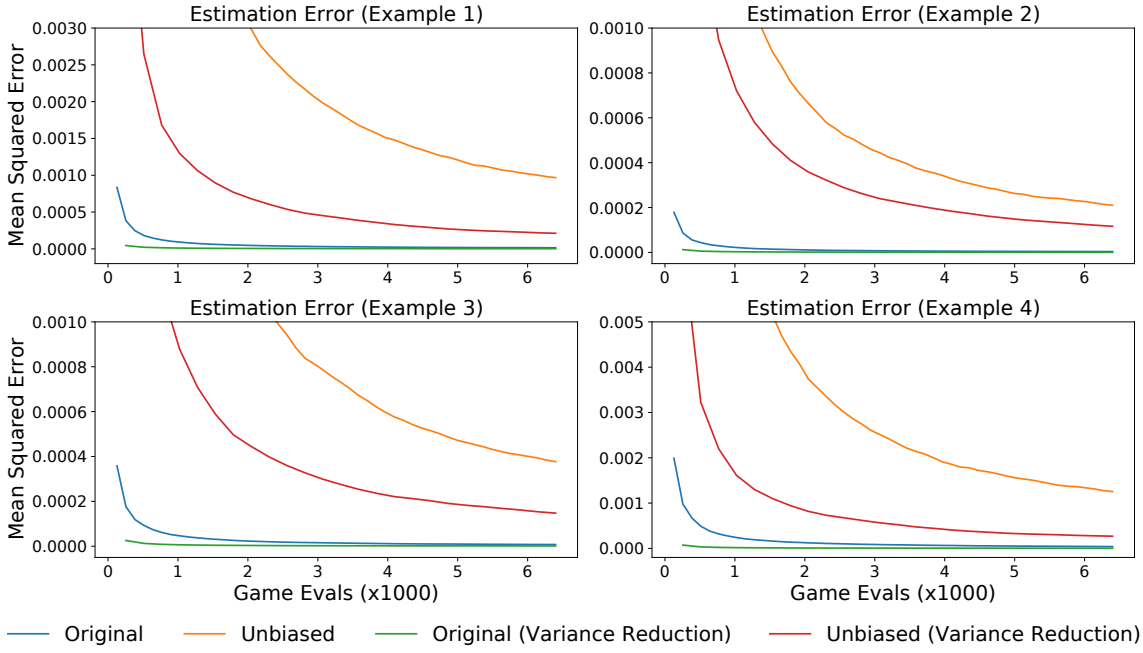

Figure 1: Census income SHAP value estimation error on four predictions.

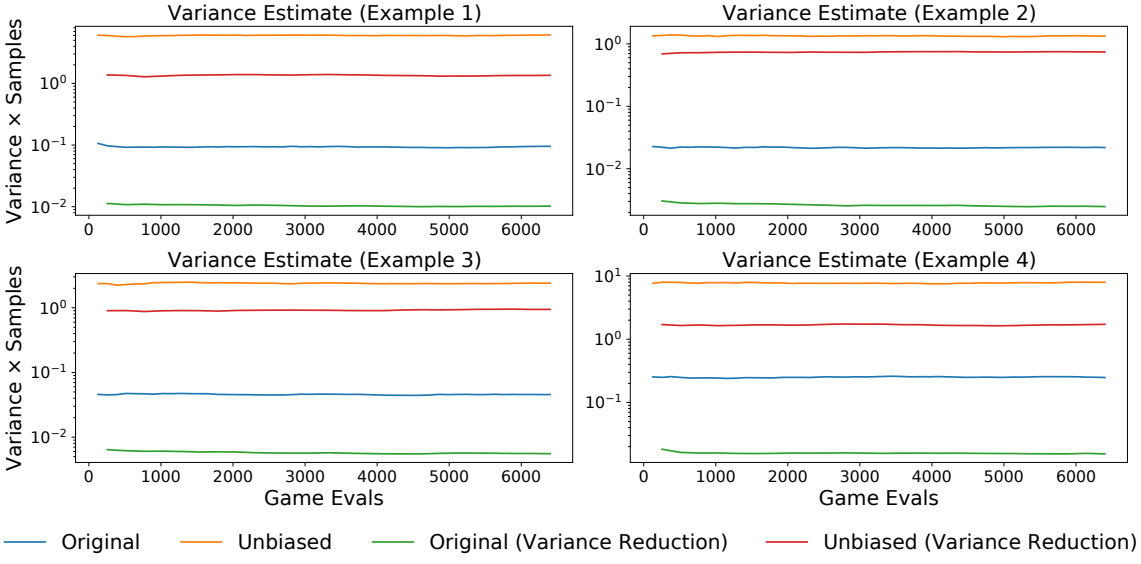

Figure 2: Census income SHAP value variance estimation on four predictions.

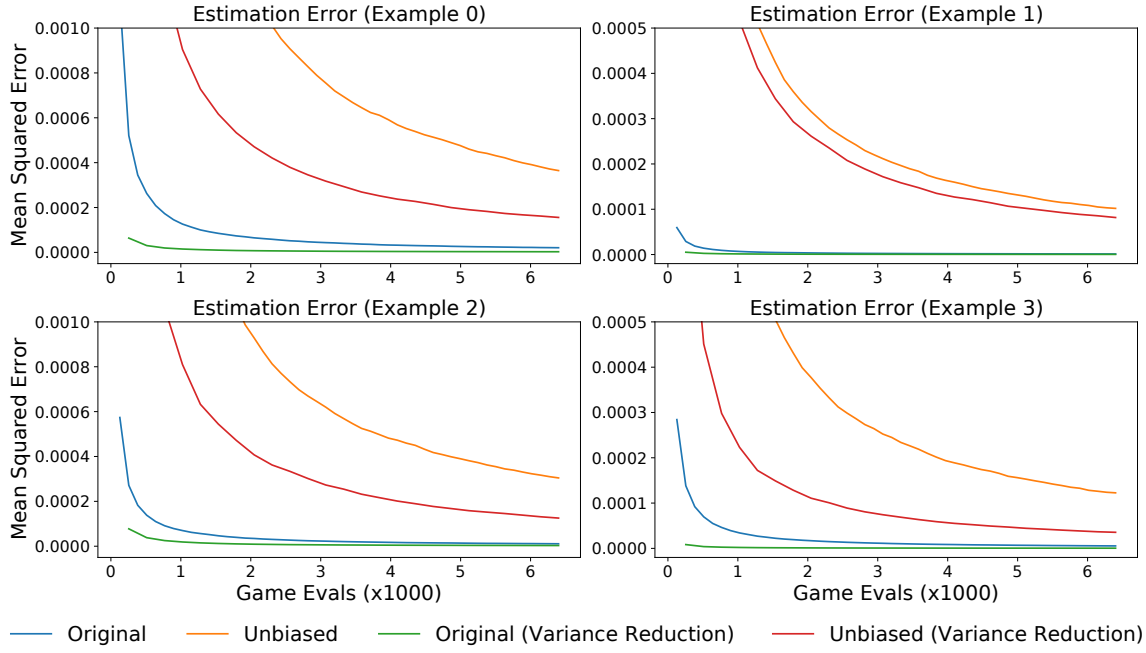

Figure 3: Bank marketing SHAP value estimation error on four predictions.

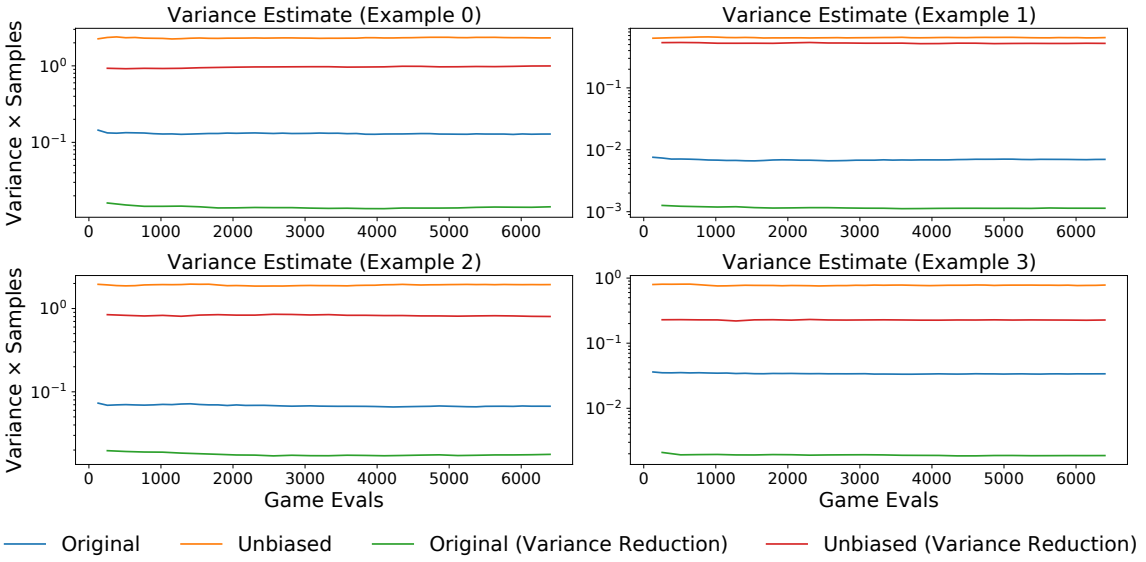

Figure 4: Bank marketing SHAP value variance estimation on four predictions.

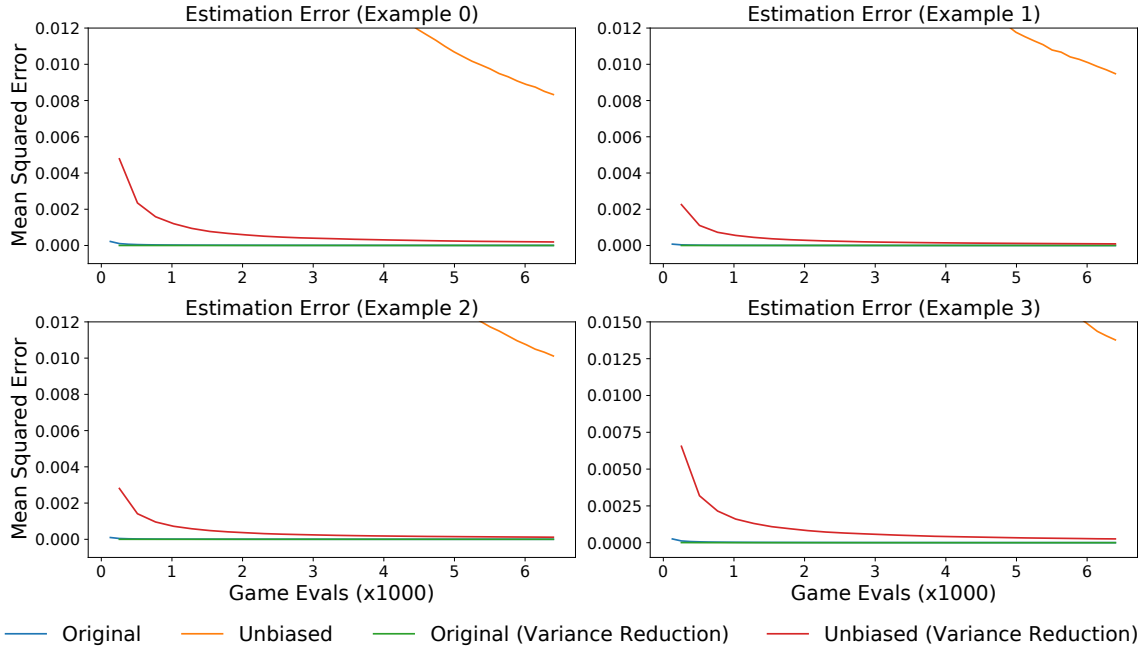

Figure 5: German credit SHAP value estimation error on four predictions.

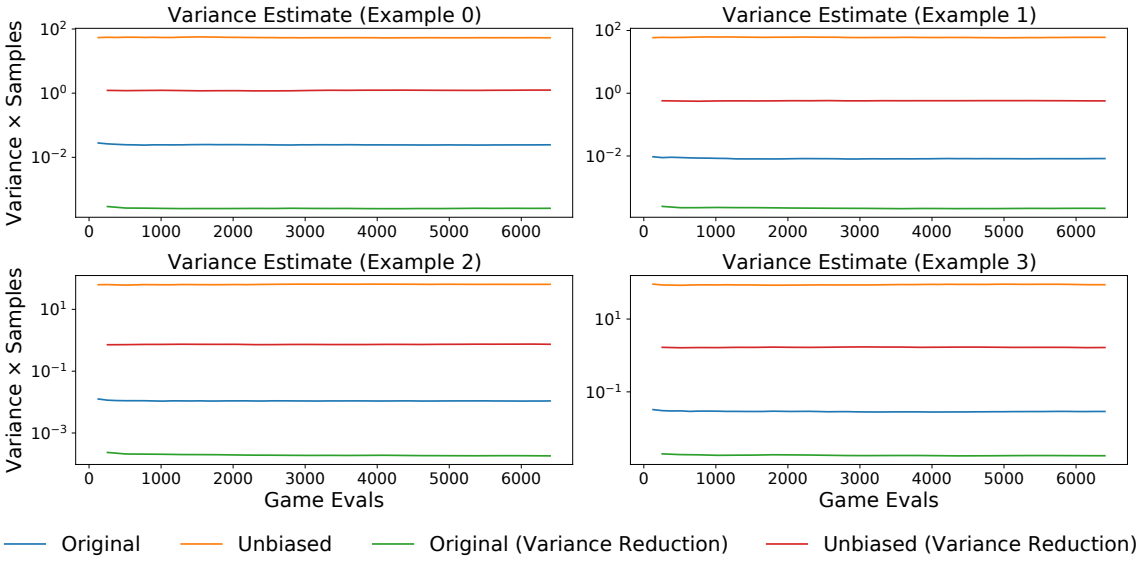

Figure 6: German credit SHAP value variance estimation on four predictions.

Table 1: Global measures of bias and variance for each SHAP value estimator. Each entry is the mean bias and mean variance calculated empirically across 100 examples (bias/variance, lower is better).

	Census Income	Bank Marketing	German Credit
Unbiased	0.0002/0.0208	0.0001/0.0125	0.0026/0.2561
Unbiased + Paired Sampling	0.0000/0.0068	0.0000/0.0066	0.0000/0.0062
Original (KernelSHAP)	0.0000/0.0007	0.0000/0.0006	0.0000/0.0002
Original + Paired Sampling	0.0000/0.0001	0.0000/0.0001	0.0000/0.0000

7 ALGORITHMS

Here, we provide pseudocode for the estimation algorithms described in the main text. Algorithm 1 shows the dataset sampling approach (original KernelSHAP) with our convergence detection and paired sampling techniques. Algorithm 2 shows KernelSHAP's adaptation to the setting of stochastic cooperative games (stochastic KernelSHAP). Algorithm 3 shows the unbiased KernelSHAP estimator, and Algorithm 4 shows the adaptation of unbiased KernelSHAP to stochastic cooperative games.

References

[1] Ian Covert, Scott Lundberg, and Su-In Lee. Understanding global feature contributions with additive importance measures. *Advances in Neural Information Processing Systems*, 34, 2020.

[2] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774, 2017.

[3] Art B Owen. Sobol' indices and Shapley value. *SIAM/ASA Journal on Uncertainty Quantification*, 2(1): 245–251, 2014.

Algorithm 1: Shapley value estimation with dataset sampling (KernelSHAP)

Input: Game v, convergence threshold t, intermediate samples m

`// Initialize`
n = 0
A = 0
b = 0

`// For tracking intermediate samples`
counter = 0
Atemp = 0
btemp = 0
estimates = list()

`// Sampling loop`
converged = False
while not converged **do**
  `// Draw next sample`
  Sample $z \sim p(Z)$
  if variance reduction **then**
    Asample = $\frac{1}{2}\big(zz^T + (\mathbf{1}-z)(\mathbf{1}-z)^T\big)$
    bsample = $\frac{1}{2}\big(zv(z) + (\mathbf{1}-z)v(\mathbf{1}-z) - v(\mathbf{0})\big)$
  else
    Asample = zz^T
    bsample = $z\big(v(z) - v(\mathbf{0})\big)$

  `// Welford's algorithm`
  n = n + 1
  A += (Asample − A) / n
  b += (bsample − b) / n
  counter += 1
  Atemp += (Asample − Atemp) / counter
  btemp += (bsample − btemp) / counter

  if counter == m **then**
    `// Get intermediate estimate`
    $\beta_m = \text{Atemp}^{-1}\Big(\text{btemp} - \mathbf{1}\frac{\mathbf{1}^T\text{Atemp}^{-1}\text{btemp} - v(\mathbf{1}) + v(\mathbf{0})}{\mathbf{1}^T\text{Atemp}^{-1}\mathbf{1}}\Big)$
    estimates.append(β_m)
    counter = 0
    Atemp = 0
    btemp = 0

    `// Get estimates, uncertainties`
    $\beta_n = \text{A}^{-1}\Big(\text{b} - \mathbf{1}\frac{\mathbf{1}^T\text{A}^{-1}\text{b} - v(\mathbf{1}) + v(\mathbf{0})}{\mathbf{1}^T\text{A}^{-1}\mathbf{1}}\Big)$
    $\Sigma_\beta = m \cdot \text{Cov}(\text{estimates})$ `// Empirical covariance`
    $\sigma_n = \sqrt{\text{diag}(\Sigma_\beta)/\text{n}}$ `// Element-wise square root`

    `// Check for convergence`
    converged = $\Big(\frac{\max(\sigma_n)}{\max(\beta_n) - \min(\beta_n)} < t\Big)$
end
return β_n, σ_n

Algorithm 2: Shapley value estimation with dataset sampling for stochastic cooperative games

Input: Game V, convergence threshold t, intermediate samples m

`// Initialize`
n = 0
A = 0
b = 0

`// For tracking intermediate samples`
counter = 0
Atemp = 0
btemp = 0
estimates = list()

`// Sampling loop`
converged = False
while not converged **do**
  `// Draw next sample`
  Sample $z \sim p(Z)$
  Sample $u \sim p(U)$
  if variance reduction **then**
    bsample = $\frac{1}{2}\big(zV(z,u) + (\mathbf{1}-z)V(\mathbf{1}-z,u) - \mathbb{E}_U[V(\mathbf{0},U)]\big)$
    Asample = $\frac{1}{2}\big(zz^T + (\mathbf{1}-z)(\mathbf{1}-z)^T\big)$
  else
    bsample = $z\big(V(z,u) - \mathbb{E}_U[V(\mathbf{0},U)]\big)$
    Asample = zz^T

  `// Welford's algorithm`
  n = n + 1
  b += (bsample − b) / n
  A += (Asample − A) / n
  counter += 1
  btemp += (bsample − btemp) / counter
  Atemp += (Asample − Atemp) / counter

  if counter == m **then**
    `// Get intermediate estimate`
    $\beta_m = \mathrm{Atemp}^{-1}\Big(\mathrm{btemp} - \mathbf{1}\frac{\mathbf{1}^T\mathrm{Atemp}^{-1}\mathrm{btemp} - \mathbb{E}_U[V(\mathbf{1},U)] + \mathbb{E}_U[V(\mathbf{0},U)]}{\mathbf{1}^T\mathrm{Atemp}^{-1}\mathbf{1}}\Big)$
    estimates.append(β_m)
    counter = 0
    Atemp = 0
    btemp = 0

    `// Get estimates, uncertainties`
    $\beta_n = \mathrm{A}^{-1}\Big(\mathrm{b} - \mathbf{1}\frac{\mathbf{1}^T\mathrm{A}^{-1}\mathrm{b} - \mathbb{E}_U[V(\mathbf{1},U)] + \mathbb{E}_U[V(\mathbf{0},U)]}{\mathbf{1}^T\mathrm{A}^{-1}\mathbf{1}}\Big)$
    $\Sigma_\beta = m \cdot \mathrm{Cov}(\text{estimates})$ `// Empirical covariance`
    $\sigma_n = \sqrt{\mathrm{diag}(\Sigma_\beta)/\mathrm{n}}$ `// Element-wise square root`

    `// Check for convergence`
    converged = $\Big(\frac{\max(\sigma_n)}{\max(\beta_n) - \min(\beta_n)} < t\Big)$
end
return β_n, σ_n

Algorithm 3: Unbiased Shapley value estimation

Input: Game v, convergence threshold t

`// Initialize`
Set A (Section 3.3)
Set C (Eq. 13)
n = 0
b = 0
bSSQ = 0

`// Sampling loop`
converged = False
while not converged **do**
  `// Draw next sample`
  Sample $z \sim p(Z)$
  if variance reduction **then**
    bsample = $\frac{1}{2}\big(zv(z) + (\mathbf{1}-z)v(\mathbf{1}-z) - v(\mathbf{0})\big)$
  else
    bsample = $zv(z) - \frac{1}{2}v(\mathbf{0})$

  `// Welford's algorithm`
  n = n + 1
  diff = (bsample − b)
  b += diff / n
  diff2 = (bsample − b)
  bSSQ += outer(diff, diff2) `// Outer product`

  `// Get estimates, uncertainties`
  $\beta_n = A^{-1}\Big(\mathrm{b} - \mathbf{1}\frac{\mathbf{1}^T A^{-1}\mathrm{b} - v(\mathbf{1}) + v(\mathbf{0})}{\mathbf{1}^T A^{-1}\mathbf{1}}\Big)$
  Σ_b = bSSQ / n
  $\Sigma_\beta = C\Sigma_b C^T$
  $\sigma_n = \sqrt{\mathrm{diag}(\Sigma_\beta)/\mathrm{n}}$ `// Element-wise square root`

  `// Check for convergence`
  converged = $\left(\frac{\max(\sigma_n)}{\max(\beta_n) - \min(\beta_n)} < t\right)$
end
return β_n, σ_n

Algorithm 4: Unbiased Shapley value estimation for stochastic cooperative games

Input: Game V, convergence threshold t

`// Initialize`
Set A (Section 3.3)
Set C (Eq. 13)
n = 0
b = 0
bSSQ = 0

`// Sampling loop`
converged = False
while not converged **do**
  `// Draw next sample`
  Sample $z \sim p(Z)$
  Sample $u \sim p(U)$
  if variance reduction **then**
    bsample = $\frac{1}{2}\Big(zV(z,u) + (\mathbf{1}-z)V(\mathbf{1}-z,u) - \mathbb{E}_U\big[V(\mathbf{0}),U\big]\Big)$
  else
    bsample = $zV(z,u) - \frac{1}{2}\mathbb{E}_U[V(\mathbf{0},U)]$

  `// Welford's algorithm`
  n = n + 1
  diff = (bsample − b)
  b += diff / n
  diff2 = (bsample − b)
  bSSQ += outer(diff, diff2) `// Outer product`

  `// Get estimates, uncertainties`
  $\beta_n = A^{-1}\Big(\mathrm{b} - \mathbf{1}\frac{\mathbf{1}^T A^{-1}\mathrm{b} - \mathbb{E}_U[V(\mathbf{1},U)] + \mathbb{E}_U[V(\mathbf{0},U)]}{\mathbf{1}^T A^{-1}\mathbf{1}}\Big)$
  Σ_b = bSSQ / n
  $\Sigma_\beta = C\Sigma_b C^T$
  $\sigma_n = \sqrt{\mathrm{diag}(\Sigma_\beta)/\mathrm{n}}$ `// Element-wise square root`

  `// Check for convergence`
  converged = $\left(\frac{\max(\sigma_n)}{\max(\beta_n) - \min(\beta_n)} < t\right)$
end
return β_n, σ_n