Proceedings of the 2025 Winter Simulation Conference
E. Azar, A. Djanatliev, A. Harper, C. Kogler, V. Ramamohan, A. Anagnostou, and S. J. E. Taylor, eds.

USING ADAPTIVE BASIS SEARCH METHOD IN QUASI-REGRESSION TO INTERPRET BLACK-BOX MODELS

Ambrose Emmett-Iwaniw[1], and Christiane Lemieux[1]

[1]Dept. of Statistics and Actuarial Science, University of Waterloo, Waterloo, Ontario, Canada.

ABSTRACT

Quasi-Regression (QR) is an inference method that approximates a function of interest (e.g., black-box model) for interpretation purposes by a linear combination of orthonormal basis functions of $L^2[0,1]^d$. The coefficients are integrals that do not have an analytical solution and therefore must be estimated, using Monte Carlo or Randomized Quasi-Monte Carlo (RQMC). The QR method can be time-consuming if the number of basis functions is large. If the function of interest is sparse, many of these basis functions are irrelevant and could thus be removed, but they need to be correctly identified first. We address this challenge by proposing new adaptive basis search methods based on the RQMC method that adaptively select important basis functions. These methods are shown to be much faster than previously proposed QR methods and are overall more efficient.

1 INTRODUCTION

In machine learning and statistics, understanding which variables influence predictions in black-box models remains a critical challenge (Guidotti et al. 2018; Casalicchio et al. 2019). Quasi-regression (QR) offers a method to interpret black-box models by approximating them using orthonormal basis functions (Jiang and Owen 2002). A common technique to interpret black-box models is using the SHapley Additive exPlanations (SHAP) technique (Ekanayake et al. 2022) to construct an interpretable version of the model to provide explanations for specific predictions. Another technique for interpretability is using Local Interpretable Model-agnostic Explanations (LIME) (Ribeiro et al. 2016) to approximate a black-box model locally with an interpretable model to visually explain the relative importance of all input characteristics. For further information, see the recent review by Hassija et al. (2024) on interpreting black-box models through explainable artificial intelligence. Other reasons to approximate black-box functions come from Bayesian Optimization (Garnett 2023), which uses Gaussian Processes regression (Williams and Rasmussen 2006) to approximate black-boxes to find the global optimum when the black-box evaluations are expensive. An alternative to Gaussian Process regression to approximate the black-box function is to use a polynomial surrogate instead to find the minimum of the black-box function. This is called Polynomial-Model-Based Optimization (PMBO) (Schreiber et al. 2024). This is similar to the quasi-regression approach proposed by Jiang and Owen (2002), but where the end use is optimization rather than inference.

Traditional QR methods suffer from slow convergence and computational bottlenecks due to the many coefficients involved. To improve efficiency, we propose Adaptive Quasi-Regression (AQR), which leverages randomized quasi-Monte Carlo (RQMC) sampling to accelerate computations. Inspired by Friedman's multivariate adaptive regression splines (MARS) method (Friedman 1991), our approach dynamically selects and prunes basis functions, significantly reducing runtime while maintaining accuracy. Empirical results show that AQR outperforms QR with either RQMC or shrinkage in terms of efficiency performance.

This paper is organized as follows: Section 2 provides background on QMC/RQMC and QR; Section 3 introduces our adaptive basis search method and a convergence result; Section 4 outlines the evaluation framework; Section 5 presents numerical experiments; and Section 6 concludes with key findings.

2 BACKGROUND

2.1 Quasi-Monte Carlo (QMC) and Randomized quasi-Monte Carlo (RQMC)

The QMC method replaces independent uniform random points with a low-discrepancy point set $P_n = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, which is distributed more evenly over $[0,1]^d$. It then uses P_n to evaluate $\mu = \int_{[0,1]^d} f(\mathbf{x})d\mathbf{x}$ via $\hat{\mu}_n = \sum_{i=1}^n f(\mathbf{x}_i)/n$. A key measure of uniformity is the star discrepancy $D^*(P_n)$ (Niederreiter 1992), which measures the distance between the uniform distribution and the empirical distribution induced by the point set P_n.

For QMC-based numerical integration, the Koksma-Hlawka inequality provides an error bound: $|\mu - \hat{\mu}_n| \leq V(f)D^*(P_n)$, where $V(f)$ measures function smoothness (Niederreiter 1992). However, this bound is often difficult to compute or too loose to be practical.

Certain point set constructions such as digital nets and sequences achieve $D^*(P_n) = O(n^{-1}\log(n)^{d-1})$ (L'Ecuyer 2018), outperforming the MC integration error rate of $O(n^{-1/2})$, making QMC advantageous for large n. In this paper, we use Sobol' sequences (Sobol' 1967), which are computationally efficient due to their base-2 construction (L'Ecuyer 2018). See L'Ecuyer and Lemieux (2002) for more details.

A challenge with QMC is the lack of sample-based error estimates due to its deterministic nature. A solution is RQMC, which preserves low-discrepancy properties while allowing unbiased estimation and standard error computation via replication. Various randomization techniques exist, such as digital b-ary shift and scrambling (L'Ecuyer and Lemieux 2002; Owen 1997a).

2.2 Quasi-Regression (QR)

Quasi-Regression (QR) was introduced by Owen (2000) to assess the linearity of high-dimensional functions. An and Owen (2001) later evaluated QR's quality of fit using real-world problems, measuring accuracy via the Lack-of-Fit (LOF) metric, which quantifies the unexplained variance in f.

QR derives its name from "ignoring the denominator" (Chui and Diamond 1987). It decomposes square-integrable functions $f : [0,1]^d \to \mathbb{R}$ into an infinite weighted sum of orthonormal basis functions $\{\psi_\mathbf{r}\}_{\mathbf{r}\in U}$ of $L^2[0,1]^d$, where U is an infinite index set. That is, we write $f(\mathbf{x}) = \sum_{\mathbf{r}\in U} \beta_\mathbf{r} \psi_\mathbf{r}(\mathbf{x})$, where $\beta_\mathbf{r} = \int_{[0,1]^d} f(\mathbf{x})\psi_\mathbf{r}(\mathbf{x})d\mathbf{x}$. In practice, this sum is truncated to a finite approximation, i.e., $f(\mathbf{x}) \approx \sum_{\mathbf{r}\in R} \beta_\mathbf{r} \psi_\mathbf{r}(\mathbf{x})$, where R is a finite index set of size p.

Instead of using a full least squares approach to find the coefficients $\beta_\mathbf{r}$ that minimize the error function $f(\mathbf{x}) - \sum_{\mathbf{r}\in R} \beta_\mathbf{r} \psi_\mathbf{r}(\mathbf{x})$, QR leverages the above integral formulation for $\beta_\mathbf{r}$ to reduce computational complexity from $O(np^2 + p^3)$ to $O(np)$. That is, the QR estimate of f is: $\hat{f}_n(\mathbf{x}) = \sum_{\mathbf{r}\in R} \hat{\beta}_{\mathbf{r},n} \psi_\mathbf{r}(x)$, where the unbiased estimator for $\beta_\mathbf{r}$ is $\hat{\beta}_{\mathbf{r},n} = \frac{1}{n}\sum_{i=1}^n \psi_\mathbf{r}(\mathbf{x}_i)f(\mathbf{x}_i)$.

QR uses tensor-product basis functions, often orthonormal polynomials such as transformed Legendre polynomials. That is, the basis functions have the form $\psi_\mathbf{r} = \prod_{i=1}^d \phi_{i,r_i}(x_i)$, where $\phi_{i,r_i}(x_i) : [0,1] \to \mathbb{R}$ is a polynomial of degree r_i. The index set R is typically chosen as $R := R_{B_0,B_1,B_\infty} = \{\mathbf{r} \in \mathbb{N}^d : ||\mathbf{r}||_0 \leq B_0, ||\mathbf{r}||_1 \leq B_1, ||\mathbf{r}||_\infty \leq B_\infty\}$, where the rank $||\mathbf{r}||_0$ describes the number of variables on which ψ_r depends, $||\mathbf{r}||_1$ is the degree of $\psi_\mathbf{r}$ and the order $||\mathbf{r}||_\infty$ is the maximum degree of the univariate polynomials $\phi_{i,r_i}(x)$. In practice, B_0, B_1, and B_∞ are chosen so that the size p of R remains manageable while capturing enough of the variability of f. Examples of common values are given in Section 5. For a fixed computational budget, one needs to balance between increasing n or p: increasing n reduces the variance of the $\hat{\beta}_{\mathbf{r},n}$ while increasing p reduces the bias caused by truncating U to R.

The accuracy of QR is assessed via Integrated Squared Error (ISE): $\text{ISE}(n) = \int_{[0,1]^d} (f(\mathbf{x}) - \hat{f}_n(\mathbf{x}))^2 d\mathbf{x}$. Normalized by f's variance σ^2, this defines the Lack-of-Fit (LOF): $\text{LOF}(n) = \text{ISE}(n)/\sigma^2$. LOF represents the unexplained fraction of σ^2, exceeding 1 when QR performs worse than a mean-based model.

QR helps analyze functions that may not be explicitly known, answering questions about linearity, variable importance, and interactions. Even for complex functions such as black-box models, QR provides insights into the structure and significance of terms.

3 ADAPTIVE QUASI-REGRESSION

In QR literature, the interpretable set R is assumed to be fixed and sufficiently small for feasible computation. QR approximation works well when f primarily consists of low-order interactions of low-degree univariate basis functions (An and Owen 2001; Jiang 2003; Jiang and Owen 2003). This section addresses two key issues in QR methods.

The first issue, noted in Jiang (2003), is a large truncation error when significant variability in f lies outside R. Expanding R mitigates this but becomes computationally infeasible, requiring adaptive basis selection. Jiang (2003) introduced an adaptive method improving runtime and LOF estimates. In our AQR methods, we can take advantage of RQMC to speed up convergence and reduce variance. Živanović (2021) similarly proposed adding basis terms independently when the initial set poorly approximates the function. Our AQR methods leverage the fact that we can fit coefficients independently, starting with the main effects first, and then the two-factor effects that have one of the important main effects as one of their factors, etc. This yields faster runtimes than non-adaptive QR, as we are computing fewer coefficients.

The second issue arises when the QR approximation is sparse, causing the method to learn insignificant variables, increasing runtime. By sparse, one means for a given index set R, only a small portion of the $\beta_{\mathbf{r}}, \mathbf{r} \in R$ are non-zero. Our proposed AQR methods address this problem by prioritizing key coefficients deemed effective, and accept minor truncation errors for feasibility. Živanović (2021) suggested removing small basis functions post-fitting, an idea we utilize in Section 3.1 for the CutCriteria function. Unlike Živanović (2021), which trims after fitting all coefficients, our method trims in stages: first, we trim after fitting all main effects, second, we trim after fitting two-factor coefficients with one of the terms being a main effect term, and continue until we reach the maximum interaction size B_0. This means that we fit fewer terms, increasing efficiency. Similarly, our approach differs from Jiang (2003), which updates coefficients dynamically based on Sobol' sensitivity indices.

AQR methods assume the interpretable set is sparse, aligning with the "bet on sparsity" principle (Hastie et al. 2009; Hastie et al. 2015). If f is densely represented in R, AQR provides minimal improvement and fails to produce a feasible approximation over traditional QR methods.

3.1 Methods

This section introduces three AQR methods: **AQRNNS** (Adaptive Quasi-Regression Normal Noise Shrinkage); **AQRSA** (Adaptive Quasi-Regression Sensitivity Analysis); and **AQROS** (Adaptive Quasi-Regression Optimal Shrinkage). They all use RQMC to approximate the coefficients $\beta_{\mathbf{r}}$.

These methods are inspired by Friedman's MARS method (Friedman 1991), which selects basis functions through a forward and backward pass. The AQR methods follow a similar approach, starting by fitting main effects and then progressively incorporating higher-order interactions. We first fit main effects using a point set of size n, and then we apply a cut-off procedure to remove noisy or insignificant coefficients. After this, we fit all possible two-factor interactions that have a factor that has a significant main effect and apply the cut-off procedure again. We repeat for three-factor and higher-order interactions. To be clear, our procedure is such that we cannot repeatedly select and drop a coefficient.

We first present the general form of the AQR methods, and later explain how each of them execute the cut-off procedure. In the pseudocode below, we assume X is the k^{th} realization of a RQMC point set of size n, also denoted $\tilde{P}_{n,k}$ later on. The user must also choose the values B_0, B_1 and B_∞ defining R.

In the above pseudocode, the helper function `FitBeta` estimates a coefficient and its biased variance for a given index using n RQMC samples X and corresponding function values $f(X)$. The helper function `FitBetaInactive` loops through each index in IndexSet, calls FitBeta and returns all fitted beta coefficients and biased variance estimates whose index is in IndexSet. Finally, `PotentInterPolyFcts` returns all the next order of interaction terms for the given Index, assuming the upper bound on the degree is at most B_1 and the upper bound on the order is at most B_∞.

```
Generic AQR
    Input: $n$, $B_0$, $B_1$, $B_\infty$, $X$, $f(X)$, Optional $\alpha$.
    Output: activeSet (the selected important basis functions interpretable indices.
activeSet = $\emptyset$ // Contains all the important basis functions interpretable indices.
activeBetas = $\emptyset$ // Contains all the important basis functions, coefficients and biased variance values.
// Fit the intercept.
interceptIndex = $\mathbf{0}_{d\times 1}$
activeSet = activeSet $\cup$ {interceptIndex}
activeBetas = activeBetas $\cup$ {FitBeta(n, interceptIndex, $X$, $f(X)$)}
// $i$ stands for the current interaction that will be extended to $i+1$ interaction.
// For example, $i=0$ intercept and extending to $i=1$ main effect terms.
for $i=0,\cdots,B_0-1$ do
    newActiveSet = $\emptyset$ // Contains all the new important basis functions interpretable indices.
    newActiveBetas = $\emptyset$ // Contains all the new important basis functions coefficient/variance values.
    activeSetSize = size of activeSet
    // This loops through each term in the activeSet that has a current rank of $i$
    // and adds new extended indices that have at most order $B_1$ and degree $B_\infty$.
    for $j=0,\cdots,$ activeSetSize - 1 do
        if $||\text{activeSet}[j]||_0 == i$ then
            newActiveSet = PotentInterPolyFcts(activeSet[j], $B_1$, $B_\infty$)
            newActiveSet = newActiveSet \ activeSet // Difference of sets i.e,. removes duplicates.
            newBetasActive = FitBetaInactive($n$, newActiveSet, $X$, $f(X)$)
            activeSet = activeSet $\cup$ newActiveSet
            activeBetas = activeBetas $\cup$ newActiveBetas
        end if
    end for
    activeSet, activeBetas = CutCriteria(activeSet, activeBetas, $X$, $f(X)$, $\alpha$) // See Algorithms 1–3 for details on implementing CutCriteria.
end for
return activeSet
```

We now describe the three different variants we propose for the cut-off procedure.

Algorithm 1: Adaptive Quasi-Regression Normal Noise Shrinkage (AQRNNS)

For the AQRNNS method, the idea of the cut-off procedure comes from classical linear regression. For each coefficient, using the α significance level, assuming normal errors, a $1-\alpha$ confidence interval is created around the value of the coefficient being zero. If zero is contained in the confidence interval, the coefficient is likely small and is removed.

```
Function CutCriteriaAQRNNS(IndexSet, BetaSet, $X$, $Y$, $\alpha$)
    Output: IndexSet (the selected important basis indices), BetaSet (the selected important basis coefficients and biased variances).
activeSize = size of IndexSet
$z=\Phi^{-1}(1-\alpha/2)$
for $k=\text{activeSize}-1,\cdots,0$ do
    $\beta_k=\text{BetaSet}[k][0], \text{Var}(\beta_k)=\text{BetaSet}[k][1]$
    if $|\beta_k|<z\sqrt{\text{Var}(\beta_k)}$ then
        Remove the $k^{th}$ element from IndexSet.
    end if
end for
return IndexSet, BetaSet[IndexSet]
EndFunction
```

Algorithm 2: Adaptive Quasi-Regression Optimal Shrinkage (AQROS)

The AQROS algorithm uses a similar idea to classical linear regression, but instead of using the normal quantile, it uses the threshold (1) for wavelet shrinkage (Donoho and Johnstone 1994), given by:

$$\sqrt{2\log(p)\widehat{\text{Var}}(\hat{\beta}_{\mathbf{r},n})}. \tag{1}$$

This threshold (1) is chosen as there is no need for a cut-off parameter. Jiang (2003) utilize it in their shrinkage method showing promise numerically when the function of interest is sparse (i.e., most of the coefficients are equal to 0).

```
Function CutCriteriaAQROS(IndexSet, BetaSet, $X$, $Y$, $\alpha$)
    Output: IndexSet (the selected important basis indices), BetaSet (the selected important basis coefficients and biased variances).
activeSize = size of IndexSet
for $k = \text{activeSize} - 1, \cdots, 0$ do
    $\beta_k = \text{BetaSet}[k][0], \text{Var}(\beta_k) = \text{BetaSet}[k][1]$
    if $|\text{BetaSet}[k]| < \sqrt{2\log(\text{activeSize})\text{Var}(\beta_k)}$ then
        Remove the $k^{th}$ element from IndexSet.
    end if
end for
return IndexSet, BetaSet[IndexSet]
EndFunction
```

Algorithm 3: Adaptive Quasi-Regression Sensitivity Analysis (AQRSA)
The AQRSA algorithm cut-off procedure keeps collecting the largest coefficients whose squared values summed together and divided by the total variability (hence the name sensitivity analysis) is greater than $1-\alpha$. This essentially removes the coefficients that explain very little of the variability.

```
Function CutCriteriaAQRSA(IndexSet, BetaSet, $X$, $Y$, $\alpha$)
    Output: importantIndices (the selected important basis indices), importantBetas (the selected important basis coefficients).
importantBetas = $\emptyset$; importantIndices = $\emptyset$
importantBetas = importantBetas $\cup$ {BetaSet[0]}
importantIndices = importantIndices $\cup$ {IndexSet[0]}
BetaSetSize = size of BetaSet
// Note: Here we do not need to compute biased variance.
totalVariability = $\sum_{k=1}^{\text{BetaSetSize}} \text{BetaSet}[k]^2$
squaredBetas = store the squared elements in BetaSet
sortedIndices = store indices of elements of squaredBetas sorted in descending order
currentVarability = 0
// Skip $k = 0$ as this is the intercept term.
for $k = 1, \cdots, \text{IndexSet}.size() - 1$ do
    currentVarability += squaredBetas[sortedIndices[k]]
    importantBetas = importantBetas $\cup$ {BetaSet[sortedIndices[k]]}
    importantIndices = importantIndices $\cup$ {IndexSet[sortedIndices[k]]}
    if currentVarability / totalVariability $> 1 - \alpha$ break end if
end for
return importantIndices, importantBetas
EndFunction
```

3.2 Choice of the Cut-Off Parameter α

Ideally, the cut-off parameter α necessary to apply our AQRNNS and AQRSA methods should be chosen to minimize LOF. Since the ideal α is unknown, we use the following ad hoc procedure to select it: we run a small number of pilot runs across ten potential values for α and then choose the value with the lowest LOF. Our second method AQROS has the advantage of not requiring this extra step.

3.3 Use of Quasi-Monte Carlo

As previously indicated, in both the simple (non-adaptive) QR method and the AQR ones, we use RQMC sampling and refer to the corresponding simple QR method as RQR in this case. The expected value of the LOF is expected to be smaller when using RQMC than MC, assuming the bias caused by truncating to basis functions in R is dominated by the variance of the approximation $\hat{f}_n$. In the simplistic ideal case where f can be represented exactly with a finite R, we have the following result.

Proposition 1 Assume $f(\mathbf{x})$ is of the form $f(\mathbf{x}) = \sum_{\mathbf{r}\in R} \beta_{\mathbf{r}} \varphi_{\mathbf{r}}(\mathbf{x})$ for some finite set R and uses transformed Legendre polynomials as the basis functions. Let P_n be obtained as a scrambled digital net with $LOF(n)$ for the RQR approximation denoted as $LOF_{rqmc}(n)$. $\mathrm{E}(LOF_{rqmc}(n)) = O((\log n)^d/n^3)$.

Proof. Given the assumption on f, we have that $\mathrm{E}(LOF_{rqmc}(n)) = \sum_{\mathbf{r}\in R} \mathrm{E}((\beta_{\mathbf{r}} - \hat{\beta}_{\mathbf{r}})^2)/\sum_{\mathbf{r}\in R \neq \mathbf{0}} \beta_{\mathbf{r}}^2$ where each $\hat{\beta}_{\mathbf{r}}$ is the unbiased RQMC estimator for $\beta_{\mathbf{r}}$. It can be shown that $g_{\mathbf{r}}(\mathbf{x}) = f(\mathbf{x})\varphi_{\mathbf{r}}(\mathbf{x})$ is smooth, as per Def. 2 in Owen (1997b), so we can apply Theorem 2 from Owen (1997b) which implies $\mathrm{Var}(\hat{\beta}_{\mathbf{r}}) = \mathrm{E}((\beta_{\mathbf{r}} - \hat{\beta}_{\mathbf{r}})^2) = O((\log^d n)/n^3)$ for each $\mathbf{r}$, and thus $\mathrm{E}(LOF_{rqmc}(n)) = O((\log^d n)/n^3)$. $\square$

4 FRAMEWORK FOR COMPARING METHODS

We now describe the metrics we use to compare our proposed AQR methods with other QR-based methods. The first three metrics listed below assess function approximation quality, while the next three focus on tasks related to sensitivity analysis. To expand further on the latter point, QR is particularly useful for inference when combined with functional ANOVA decompositions, which help quantify the variability explained by different effects, leading to global sensitivity indices. These indices highlight influential variables or interactions, often computed using Monte Carlo methods (Sobol' 1990; Sobol 2001). They can be useful for interpreting a black-box model, as shown in Jiang and Owen (2002).

Functional ANOVA (Hoeffding 1948) decomposes a square-integrable function f into 2^d orthogonal component functions f_u, each corresponding to a subset u of input variables. The variance σ_u^2 of each term f_u measures its contribution to the total variability of f, and the global sensitivity index $S(u) = \sigma_u^2/\sigma^2$ quantifies its relative importance. These indices, often called Sobol' indices, distinguish main effects ($|u| = 1$) from interactions ($|u| > 1$).

A function has an effective dimension s in the superposition sense if most of its variance ($\geq 99\%$) comes from subsets of size s or smaller (Caflisch, Morokoff, and Owen 1997). QR methods, as used in Lemieux and Owen (2002), help analyze effective dimension by leveraging the identity $\sigma_u^2 = \sum_{\mathbf{r}\in S_u} \beta_{\mathbf{r}}^2$, where $S_u = \{\mathbf{r} \in \mathbb{N}^d : r_j > 0 \Leftrightarrow j \in u\}, u \subset \{1, \cdots, d\}$. The concept of effective dimension is crucial to understand the effectiveness of QMC methods. Functions with low effective dimension benefit from QMC's evenly distributed point sets, reducing integration errors (Owen 1997a).

Notation

We analyze performance as the number of evaluation points n increases as a power of 2 until n reaches a pre-fixed maximal value denoted N. Let $\tilde{\beta}_{n,\mathbf{r},k}$ be the estimated coefficient for $\beta_{\mathbf{r}}$ for a QR method using the first n points of the k^{th} replication of a d-dimensional RQMC point set $\tilde{P}_{N,k}$ of size N (except for QRS, where we use i.i.d. uniforms instead), and let $\tilde{f}_{n,k}(\mathbf{x})$ denote the corresponding function approximation.

Lack-Of-Fit (LOF)

As indicated in Section 2.2, LOF quantifies approximation accuracy, with lower values indicating better fit. Since the exact LOF cannot be computed, MC estimation is used. We estimate LOF using $m = 5000$, i.i.d. uniform samples of d dimension with Q replications (i.e., $\{\mathbf{z}_{\ell,k}\}_{\ell=1,\cdots,m,k=1,\cdots,Q} \overset{i.i.d}{\sim} U(0,1)^d$) at values of n that are powers of 2:

$\widehat{LOF}(n) = \frac{1}{Q}\sum_{k=1}^{Q} \widehat{ISE}_k(n)/\hat{\sigma}_k^2$,

where $\widehat{ISE}_k(n) = \frac{1}{m}\sum_{i=1}^{m}(f(\mathbf{z}_{i,k}) - \tilde{f}_{n,k}(\mathbf{z}_{i.k}))^2$ is the estimated integrated squared error and $\hat{\sigma}_k^2 = \sum_{i=1}^{m}(f(z_{i,k}) - \frac{1}{n}\sum_{j=1}^{m} f(z_{j,k}))^2/(m-1)$ is the unbiased sample variance based on the $\mathbf{z}_{\ell,k}$.

Standard error is: $\widehat{LOF}(n)_{SE} = \sqrt{\frac{1}{Q(Q-1)}\sum_{k=1}^{Q}\left(\widehat{ISE}_k(n)/\hat{\sigma}_k^2 - \widehat{LOF}(n)\right)^2}$.

Efficiency Metrics

Efficiency LOF measures method efficiency: Efficiency LOF(n) $= (\text{Run time} \times \widehat{LOF}(n))^{-1}$.

Convergence Rates

Convergence rates indicate how quickly methods improve with increasing n. The rate is computed as the slope α in a log-log regression of estimated LOF values, and is provided on Figures 5.1 and 5.2.

Sensitivity Measures

As seen in this section, global sensitivity indices quantify variability explained by effects: $\widehat{S(u)} = \frac{1}{Q}\sum_{k=1}^{Q}\widehat{S}_k(u)$,

where $\widehat{S}_k(u) = \sum_{\mathbf{r}\in S_{u,R}}\tilde{\beta}^2_{\mathbf{r},k}/\sum_{\mathbf{r}\in R,\mathbf{r}\neq\mathbf{0}}\tilde{\beta}^2_{\mathbf{r},k}$ with $S_{u,R} = \{\mathbf{r}\in R | r_j > 0 \Leftrightarrow j \in u\}$, $u \subset \{1,2,\cdots,d\}$ measures the contribution from subset u for the k^{th} replication.
The standard error is: $\widehat{S(u)}_{SE} = \sqrt{\frac{1}{Q(Q-1)}\sum_{k=1}^{Q}(\widehat{S}_k(u) - \widehat{S(u)})^2}$.

Effective Dimension

The effective dimension quantifies the minimum interaction size $|u|$ needed to explain 99% of the variance. Lower values improve RQMC effectiveness.

Average Interpretable Size

For AQR methods, the size of the interpretable sets R varies across replications, so we report the average over Q replications.

5 NUMERICAL EXPERIMENTS

We present three examples demonstrating how AQR methods improve runtime and reduce LOF estimates compared to RQR and QR with shrinkage (QRS). This method was first introduced by Jiang and Owen (2003) as a generalized version of centered QR, studied in Owen (2000) and Lemieux and Owen (2002). It consists in using the coefficients $\beta_{\mathbf{r}}$ as control variates with multiplicative shrinkage parameters between 0 and 1, of those used in wavelet smoothing. The estimated QR coefficients are shrunk towards zero, by these shrinkage coefficients. This reduces the variability in the estimation of the coefficients. These shrinkage strategies provide better accuracy than ordinary QR (Jiang and Owen 2003). See Jiang and Owen (2003) and Jiang (2003) for more details on QRS.

For each example, we compare the three AQR methods in Section 3 with RQR and QRS. While QRS achieves low LOF estimates, it suffers from long runtimes with large interpretable sets.

First, using the artificial sparse function from Jiang (2003), we demonstrate that AQR efficiently selects important variables with faster runtimes and lower LOF.

Second, we evaluate a neural network trained on the CPU dataset from Venables and Ripley (2013), showing that AQR approximates the black-box model more quickly than both RQR and QRS methods while maintaining similar LOF and variable importance.

Third, using a neural network trained on the Breast Cancer dataset from Street et al. (1993), we illustrate AQR's practicality in high-dimensional problems. AQR significantly outperforms QRS and RQR in runtime while achieving comparable Sensitivity Indices and LOF estimates.

For these examples, we used Q replications, generated via Parallel Processing with the **parallel** R package, leveraging the system's maximum threads as recommended by L'Ecuyer (2018). To enhance speed, we utilized the **Rcpp** package (Eddelbuettel and François 2011) for R and C++ integration and the **qrng** package (Hofert, M. and Lemieux, C. 2023) for generating digitally-shifted Sobol' sequences. In the tables, the two sections below the LOF estimate provide information on sensitivity measures $\hat{S}(u)$ for main and some higher-order effects. The run time (in seconds) is for running a given QR method over Q replications and a total of N samples.

5.1 Artificial Sparse Function

The artificial sparse function from Jiang and Owen (2003) is $f(\mathbf{x}) = \sum_{\mathbf{r}\in R_a} 0.8^{||\mathbf{r}||_1} \psi_{\mathbf{r}}(\mathbf{x})$, where $d = 8$ is the input dimension and $\{\psi_{\mathbf{r}}\}_{\mathbf{r}\in R_a}$ are the tensor products of orthonormal transformed legendre polynomials (An and Owen 2001). The actual interpretable set R_a is defined as $R_a = \{\mathbf{r} : ||\mathbf{r}||_0 \leq 2, ||\mathbf{r}||_1 \leq 2, ||\mathbf{r}||_\infty \leq 2\}$ with a size of 45. All the QR methods are performed on the following interpretable set $R = \{\mathbf{r} : ||\mathbf{r}||_0 \leq 3, ||\mathbf{r}||_1 \leq 4, ||\mathbf{r}||_\infty \leq 3\}$ with a size of 417. We choose the α parameter for the AQRNNS and AQRSA methods are 0.05 and 0.02, respectively. We use $N = 2^{17}$ total samples of dimension d.

Table 1: Results are for $Q = 20$ replications of $N = 2^{17}$ generated samples of dimension $d = 8$. All results in the table are the averages over the Q replications. In parentheses are the standard errors.

QR Methods	RQR	QRS	AQRNNS	AQRSA	AQROS
LOF	$2.02\times10^{-5}_{(3.71\times10^{-7})}$	$1.59\times10^{-5}_{(1.42\times10^{-6})}$	$6.03\times10^{-7}_{(1.6\times10^{-8})}$	$6.03\times10^{-7}_{(1.6\times10^{-8})}$	$6.03\times10^{-7}_{(1.6\times10^{-8})}$
x_1 (Similar values for $x_2,\cdots,x_8$)	$5.28\times10^{-2}_{(7.32\times10^{-6})}$	$5.28\times10^{-2}_{(7.08\times10^{-5})}$	$5.28\times10^{-2}_{(6.35\times10^{-6})}$	$5.28\times10^{-2}_{(6.35\times10^{-6})}$	$5.28\times10^{-2}_{(6.35\times10^{-6})}$
Total additive	$4.23\times10^{-1}_{(5.34\times10^{-5})}$	$4.23\times10^{-1}_{(1.21\times10^{-4})}$	$4.23\times10^{-1}_{(5.16\times10^{-5})}$	$4.23\times10^{-1}_{(5.16\times10^{-5})}$	$4.23\times10^{-1}_{(5.16\times10^{-5})}$
Total two-factor	$5.77\times10^{-1}_{(5.34\times10^{-5})}$	$5.77\times10^{-1}_{(1.21\times10^{-4})}$	$5.77\times10^{-1}_{(5.16\times10^{-5})}$	$5.77\times10^{-1}_{(5.16\times10^{-5})}$	$5.77\times10^{-1}_{(5.16\times10^{-5})}$
Total three-factor	$1.48\times10^{-5}_{(3.85\times10^{-7})}$	$1.82\times10^{-5}_{(9.19\times10^{-7})}$	$0\times10^{+00}_{(0\times10^{+00})}$	$0\times10^{+00}_{(0\times10^{+00})}$	$0\times10^{+00}_{(0\times10^{+00})}$
Interpretable Set Size	417	417	45	45	45
Run time (Seconds)	79.79	76.51	66.79	62.45	67.71
Efficiency LOF	$6.21\times10^{+02}$	$8.2\times10^{+02}$	$2.48\times10^{+04}$	$2.66\times10^{+04}$	$2.45\times10^{+04}$

As seen in Table 1, AQR methods produce lower LOF estimates than RQR and QRS because they remove small beta coefficients that are truly 0, which aligns with the intuition of Section 3. Their sensitivity matches RQR and QRS, except for a total three-factor effect of 0, as AQR removes all three-factor terms. All QR methods have an effective dimension of 2, meaning that two factors explain more than 99% of the variability. The average interpretable set of AQR is 45, less than RQR and QRS, accurately selecting the true sparse set by trimming the coefficients. This also results in faster runtimes. The AQR methods achieve higher efficiency in the LOF because of the speed and smaller LOF estimates. AQRSA is the most efficient because it skips the variance estimation for coefficients.

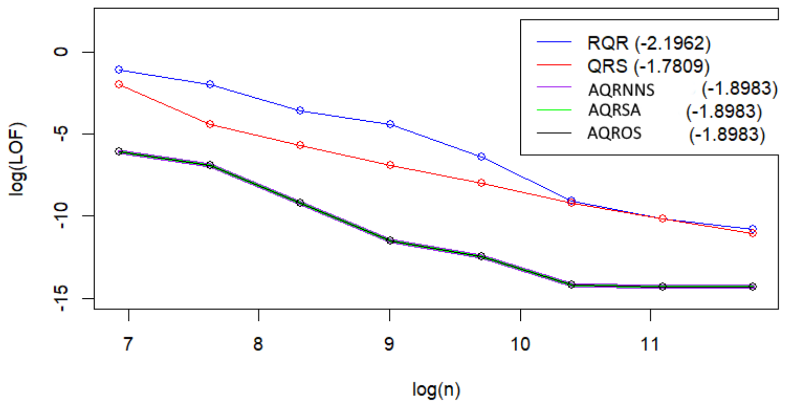

Figure 1: The LOF estimates for each QR method over the Q replications on the log-scale for $n = 2^{10}, 2^{11}, \cdots, 2^{16}, 2^{17}$. Note log(.) is of base e.

In Figure 5.1, the AQR methods (AQRNNS, AQRSA, and AQROS) exhibit faster convergence than RQR and QRS in estimating beta coefficients, particularly after $n = 2^{16}$. Their advantage comes from setting some beta coefficients to zero, aligning with their true values. However, the computed convergence rate appears slower due to the inclusion of $n = 2^{10}$ to $n = 2^{12}$ in the analysis.

In terms of LOF estimates, AQR methods consistently outperform RQR and QRS, achieving the lowest LOF values. Among them, AQRSA is the fastest, while AQROS performs well without requiring a cutoff hyperparameter. These findings highlight AQR methods' superior efficiency in QR fitting.

5.2 Training a Neural Network on the CPU Dataset

We analyze a black-box neural network trained on the CPU dataset from Venables and Ripley (2013) to predict computer CPU performance. The dataset contains 209 computers, with the response (speed) and predictors log-transformed. The predictor chmin is replaced by chmax - chmin, and all six predictors are scaled to $[0,1]^6$. The neural network has 6 input nodes, 3 hidden nodes, and 1 output node with a sigmoid activation, fitted using R's *nnet* function. Given the black-box nature of neural networks, we examine input effects and relative importance using $N = 2^{16}$ total samples of dimension $d = 6$ with Q replications. The interpretable set $R_{3,8,4}$ has size 1145. We set α to 0.05 and 0.002 for AQRNNS and AQRSA, respectively.

Table 2: Results are for $Q = 20$ replications and $N = 2^{16}$ generated samples of dimension $d = 6$. All results in the table are the averages over the Q replications. In parentheses are the standard errors.

QR Methods	RQR	QRS	AQRNNS	AQRSA	AQROS
LOF	$1.6 \times 10^{-1}_{(1.81\times10^{-3})}$	$1.63 \times 10^{-1}_{(1.69\times10^{-3})}$	$1.62 \times 10^{-1}_{(1.71\times10^{-3})}$	$1.61 \times 10^{-1}_{(1.23\times10^{-3})}$	$1.63 \times 10^{-1}_{(1\times10^{-3})}$
syct	$9.99 \times 10^{-2}_{(6.27\times10^{-5})}$	$9.94 \times 10^{-2}_{(2.43\times10^{-4})}$	$1 \times 10^{-1}_{(7.92\times10^{-5})}$	$1 \times 10^{-1}_{(7.78\times10^{-5})}$	$1 \times 10^{-1}_{(6.1\times10^{-5})}$
mmin	$5.69 \times 10^{-3}_{(4.36\times10^{-6})}$	$5.64 \times 10^{-3}_{(4.25\times10^{-5})}$	$5.71 \times 10^{-3}_{(4.7\times10^{-6})}$	$5.71 \times 10^{-3}_{(4.47\times10^{-6})}$	$5.71 \times 10^{-3}_{(3.51\times10^{-6})}$
mmax	$5.23 \times 10^{-2}_{(1.77\times10^{-5})}$	$5.25 \times 10^{-2}_{(1.92\times10^{-4})}$	$5.24 \times 10^{-2}_{(1.77\times10^{-5})}$	$5.25 \times 10^{-2}_{(1.84\times10^{-5})}$	$5.25 \times 10^{-2}_{(1.36\times10^{-5})}$
cach	$1.01 \times 10^{-1}_{(5.8\times10^{-5})}$	$1 \times 10^{-1}_{(2.02\times10^{-4})}$	$1.01 \times 10^{-1}_{(5.73\times10^{-5})}$	$1.01 \times 10^{-1}_{(6.42\times10^{-5})}$	$1.02 \times 10^{-1}_{(6.09\times10^{-5})}$
chmin	$2.66 \times 10^{-1}_{(4.21\times10^{-5})}$	$2.66 \times 10^{-1}_{(3.12\times10^{-4})}$	$2.67 \times 10^{-1}_{(4.01\times10^{-5})}$	$2.67 \times 10^{-1}_{(4.82\times10^{-5})}$	$2.68 \times 10^{-1}_{(3.35\times10^{-5})}$
chmax	$1.48 \times 10^{-2}_{(1.84\times10^{-5})}$	$1.47 \times 10^{-2}_{(7.26\times10^{-5})}$	$1.48 \times 10^{-2}_{(1.3\times10^{-5})}$	$1.48 \times 10^{-2}_{(1.06\times10^{-5})}$	$1.48 \times 10^{-2}_{(9.62\times10^{-6})}$
Total additive	$5.4 \times 10^{-1}_{(6.88\times10^{-5})}$	$5.39 \times 10^{-1}_{(4.38\times10^{-4})}$	$5.41 \times 10^{-1}_{(7.55\times10^{-5})}$	$5.42 \times 10^{-1}_{(8.43\times10^{-5})}$	$5.43 \times 10^{-1}_{(6.23\times10^{-5})}$
Total two-factor	$3.13 \times 10^{-1}_{(2.57\times10^{-5})}$	$3.12 \times 10^{-1}_{(3.89\times10^{-4})}$	$3.13 \times 10^{-1}_{(3.14\times10^{-5})}$	$3.13 \times 10^{-1}_{(4.4\times10^{-5})}$	$3.14 \times 10^{-1}_{(3.68\times10^{-5})}$
Total three-factor	$1.47 \times 10^{-1}_{(5.4\times10^{-5})}$	$1.49 \times 10^{-1}_{(3.68\times10^{-4})}$	$1.46 \times 10^{-1}_{(5.4\times10^{-5})}$	$1.46 \times 10^{-1}_{(6.16\times10^{-5})}$	$1.43 \times 10^{-1}_{(6.21\times10^{-5})}$
Interpretable Set Size	1145	1145	208	181	147
Run time (Seconds)	161.83	164.53	113.29	95.16	101.03
Efficiency LOF	3.87×10^{-2}	3.74×10^{-2}	5.45×10^{-2}	6.54×10^{-2}	6.07×10^{-2}

Looking at Table 2, the AQR methods have global sensitivity indices similar to RQR and QRS but have a smaller interpretable set size. For this trained Neural Network, the QR methods explain at least 83% of the variability. The most important main effects are the minimum number of channels, the amount of cache size in kb, and the clock time in nanoseconds. At a slight cost to the LOF metric being larger, the AQR methods run faster. This leads to the AQR methods having higher efficiency LOF than for both RQR and QRS methods. Note that our results are different than Jiang and Owen (2003) for the QRS method due to random train-test set split and the random training of the neural network.

As seen in Figure 5.2, the AQR methods all have LOF curves converging faster than RQR and QRS. At $n = 2^{16}$, all the QR methods have a small difference in LOF fit. The added benefit of AQR methods is that they are faster compared to both RQR and QRS while adding only a marginal increase in LOF estimates.

5.3 Breast Cancer Using a Neural Network

This example illustrates the effectiveness of AQR methods on large interpretable sets for black-box model interpretation. We fit a Neural Network to the Breast Cancer dataset (Street et al. 1993), where the response variable indicates whether a diagnosis is malignant or benign. The dataset contains 569 observations with $d = 30$ numeric predictors describing cell nuclei characteristics. The neural network has 30 input nodes,

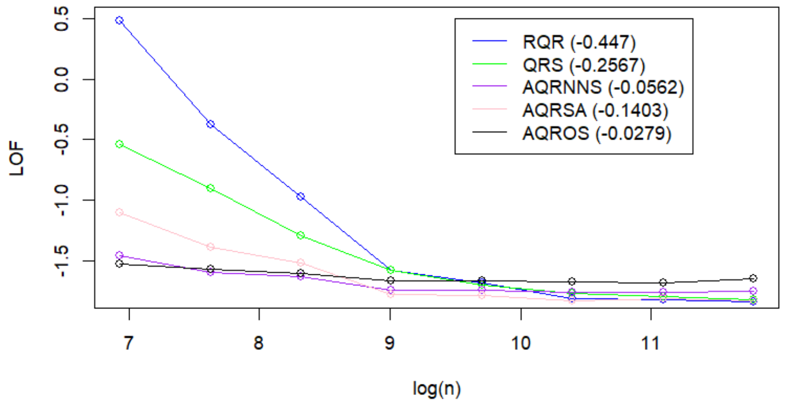

Figure 2: The LOF estimates for each QR method over the Q replications on the log-scale for the log of $n = 2^{10}, 2^{11}, \cdots, 2^{16}, 2^{17}$. Note log(.) is base e.

10 hidden nodes, and 1 output node with a sigmoid activation function. Using R's *nnet* function, data is split 80/20 for training/testing, achieving 100% training accuracy and 97.35% test accuracy.

We set $N = 2^{16}$ and compute coefficients using Q replications. The interpretable set $R_{3,8,4}$ contains 73,026 elements. AQRNNS uses $\alpha = 1^{-15}$, and AQRSA uses $\alpha = 0.001$. In this example, we used nested uniform scrambling (Owen 1997a; Owen 1997b), instead of a digital shift randomization. This was motivated by the higher dimensionality of this problem, which required the increased robustness of a scrambling-based randomization. The scrambled Sobol' sequence was generated using Art Owen's R scrambled code (Owen, A. B. 2021). Table 3 presents the simulation results.

Table 3: Results are for $Q = 20$ replications and $N = 2^{16}$ generated samples of dimension $d = 30$. All results in the table are the averages over the Q replications. In parentheses are the standard errors.

QR Methods	QRS	RQR	AQRNNS	AQRSA	AQROS
LOF estimate	$3.93 \times 10^{-3}_{(2.9\times10^{-5})}$	$1.07 \times 10^{0}_{(2.83\times10^{-2})}$	$3.12 \times 10^{-2}_{(5.75\times10^{-3})}$	$9.88 \times 10^{-3}_{(1.27\times10^{-3})}$	$8.64 \times 10^{-3}_{(6.97\times10^{-5})}$
compactness_mean	$7.5 \times 10^{-1}_{(2.32\times10^{-4})}$	$3.69 \times 10^{-1}_{(4.6\times10^{-3})}$	$7.49 \times 10^{-1}_{(3.95\times10^{-3})}$	$7.64 \times 10^{-1}_{(9.33\times10^{-4})}$	$7.69 \times 10^{-1}_{(3.25\times10^{-5})}$
smoothness_mean	$1.02 \times 10^{-1}_{(1.44\times10^{-4})}$	$5.05 \times 10^{-2}_{(6.3\times10^{-4})}$	$1.03 \times 10^{-1}_{(5.4\times10^{-4})}$	$1.05 \times 10^{-1}_{(1.28\times10^{-4})}$	$1.05 \times 10^{-1}_{(5.86\times10^{-6})}$
concave.points_worst	$2.74 \times 10^{-2}_{(6.62\times10^{-5})}$	$1.34 \times 10^{-2}_{(1.68\times10^{-4})}$	$2.73 \times 10^{-2}_{(1.42\times10^{-4})}$	$2.79 \times 10^{-2}_{(3.18\times10^{-5})}$	$2.8 \times 10^{-2}_{(4.44\times10^{-6})}$
fractal_dimension_worst	$6.1 \times 10^{-3}_{(1.71\times10^{-5})}$	$2.99 \times 10^{-3}_{(3.75\times10^{-5})}$	$6.08 \times 10^{-3}_{(3.18\times10^{-5})}$	$6.2 \times 10^{-3}_{(8.05\times10^{-6})}$	$6.24 \times 10^{-3}_{(1.45\times10^{-6})}$
perimeter_mean	$1.45 \times 10^{-3}_{(1.02\times10^{-5})}$	$7.12 \times 10^{-4}_{(8.96\times10^{-6})}$	$1.38 \times 10^{-3}_{(1.29\times10^{-5})}$	$1.25 \times 10^{-3}_{(1.7\times10^{-6})}$	$1.44 \times 10^{-3}_{(5.5\times10^{-7})}$
symmetry_worst	$9.86 \times 10^{-4}_{(6.59\times10^{-6})}$	$4.8 \times 10^{-4}_{(6.14\times10^{-6})}$	$9.75 \times 10^{-4}_{(5.2\times10^{-6})}$	$9.95 \times 10^{-4}_{(1.88\times10^{-6})}$	$1 \times 10^{-3}_{(1.3\times10^{-6})}$
area_mean	$8.41 \times 10^{-4}_{(7.64\times10^{-6})}$	$4.13 \times 10^{-4}_{(5.15\times10^{-6})}$	$7.63 \times 10^{-4}_{(4.04\times10^{-6})}$	$7.79 \times 10^{-4}_{(1.02\times10^{-6})}$	$7.84 \times 10^{-4}_{(2.44\times10^{-7})}$
area_worst	$4.58 \times 10^{-4}_{(4.78\times10^{-6})}$	$2.28 \times 10^{-4}_{(2.81\times10^{-6})}$	$4.63 \times 10^{-4}_{(2.35\times10^{-6})}$	$4.72 \times 10^{-4}_{(6.11\times10^{-7})}$	$4.75 \times 10^{-4}_{(3.72\times10^{-7})}$
perimeter_worst	$2.23 \times 10^{-4}_{(3.93\times10^{-6})}$	$1.09 \times 10^{-4}_{(1.34\times10^{-6})}$	$2.22 \times 10^{-4}_{(1.25\times10^{-6})}$	$2.26 \times 10^{-4}_{(3.53\times10^{-7})}$	$2.28 \times 10^{-4}_{(2.27\times10^{-7})}$
Total additive	$8.9 \times 10^{-1}_{(1.63\times10^{-4})}$	$4.38 \times 10^{-1}_{(5.47\times10^{-3})}$	$8.89 \times 10^{-1}_{(4.68\times10^{-3})}$	$9.06 \times 10^{-1}_{(1.11\times10^{-3})}$	$9.13 \times 10^{-1}_{(3.48\times10^{-5})}$
Total two-factor	$8.64 \times 10^{-2}_{(1.49\times10^{-4})}$	$4.41 \times 10^{-2}_{(5.17\times10^{-4})}$	$8.43 \times 10^{-2}_{(4.47\times10^{-4})}$	$8.61 \times 10^{-2}_{(1.07\times10^{-4})}$	$8.53 \times 10^{-2}_{(9.38\times10^{-6})}$
Total three-factor	$2.32 \times 10^{-3}_{(1.4\times10^{-4})}$	$5.18 \times 10^{-1}_{(5.98\times10^{-3})}$	$2.66 \times 10^{-2}_{(5.12\times10^{-3})}$	$7.66 \times 10^{-3}_{(1.21\times10^{-3})}$	$1.83 \times 10^{-3}_{(3.63\times10^{-5})}$
Effective Dimension (99%)	3	3	3	2	2
Average Interpretable Set Size	73026	73026	89	96	36
Runtime (seconds)	16956.34	16439.06	1816	881.39	825.07
Efficiency of LOF	1.5×10^{-2}	5.69×10^{-5}	1.77×10^{-2}	1.15×10^{-1}	1.4×10^{-1}

As can be seen in Table 3, for the AQR methods, all the main effects have the same ordering and are close in value to the QRS method. The effective dimension for AQRSA and AQROS is 2 as opposed to

3 for the others. This lower effective dimension improves RQMC ability to lower variance, as is seen in the standard errors. The AQR methods use a much smaller interpretable set, resulting in a significantly faster runtime. The QRS method has the lowest LOF estimate, but it takes over 15000 seconds to compute both the coefficients and the LOF estimate. At the slight cost of a larger LOF estimate, AQRNNS takes only 1816 seconds. Both AQRSA and AQROS result in a much faster runtime: less than 900 seconds to compute the coefficients and the LOF estimates. This results in AQRNNS, AQRSA and AQROS having higher efficiency of LOF. Note that the RQR method has a poor fit since many of these basis terms are not important for the fit; thus, adding several small estimated values together can become large and cause a poor LOF estimate. This identifies the need for AQR methods to remove these noisy terms, leading to a better LOF estimate with the bonus of being much faster. It is recommended to use AQR methods to investigate a sparse function when interpretable sets are large (e.g. a large number of variables). In this example, AQROS does not need to choose a cut-off parameter and is the most efficient of the AQR methods, being close to 10 times more efficient than QRS.

6 CONCLUSION

We introduce an adaptive version of QR, a fast and practical tool for evaluating variable importance for any function. These methods make QR computationally feasible in high dimensions, aided by the use of RQMC to improve convergence by lowering variance for QR in coefficient estimation. This makes AQR useful for the interpretation of both black-box and sparse functions. Identifying key variable subsets can also enhance QMC point sets for specific functions (Nuyens and Waterhouse 2012).

Note AQRNNS and AQRSA require selecting an appropriate cutoff parameter α, which, if mischosen, can degrade performance. A possible improvement is applying a prior distribution to α, turning this into a Bayesian problem akin to sparse sequential Bayesian regression (Bishop 2006).

Future work includes leveraging AQR for function approximation to optimize black-box models, as done in Bayesian Optimization (Garnett 2023) using Gaussian Processes (Williams and Rasmussen 2006).

REFERENCES

An, J., and A. Owen. 2001. "Quasi-Regression". *Journal of Complexity* 17(4):588–607.

Bishop, C. M. 2006. *Pattern Recognition and Machine Learning*. New York: Springer.

Caflisch, R. E., W. Morokoff, and A. Owen. 1997. "Valuation of Morgage Backed Securities Using Brownian Bridges to Reduce Effective Dimension". *The Journal of Computational Finance* 1:27–46.

Casalicchio, G., C. Molnar, and B. Bischl. 2019. "Visualizing the Feature Importance for Black Box Models". In *Machine Learning and Knowledge Discovery in Databases. ECML PKDD 2018. Lecture Notes in Computer Science*, edited by M. Berlingerio, F. Bonchi, T. Gärtner, N. Hurley, and G. Ifrim, Volume 11051, 655–670. Cham: Springer.

Chui, C. K., and H. Diamond. 1987. "A Natural Formulation of Quasi-Interpolation by Multivariate Splines". *Proceedings of the American Mathematical Society* 99(4):643–646.

Donoho, D. L., and I. M. Johnstone. 1994. "Ideal Spatial Adaptation by Wavelet Shrinkage". *Biometrika* 81(3):425–455.

Eddelbuettel, D., and R. François. 2011. "Rcpp: Seamless R and C++ Integration". *Journal of Statistical Software* 40:1–18.

Ekanayake, I., D. Meddage, and U. Rathnayake. 2022. "A Novel Approach to Explain the Black-Box Nature of Machine Learning in Compressive Strength Predictions of Concrete Using SHapley Additive exPlanations (SHAP)". *Case Studies in Construction Materials* 16:e01059.

Friedman, J. H. 1991. "Multivariate Adaptive Regression Splines". *The Annals of Statistics* 19(1):1–67.

Garnett, R. 2023. *Bayesian Optimization*. Cambridge: Cambridge University Press.

Guidotti, R., A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. 2018. "A Survey of Methods for Explaining Black Box Models". *ACM Computing Surveys* 51(5):1–42.

Hassija, V., V. Chamola, A. Mahapatra, A. Singal, D. Goel, K. Huang, *et al*. 2024. "Interpreting Black-Box Models: A Review on Explainable Artificial Intelligence". *Cognitive Computation* 16(1):45–74.

Hastie, T., R. Tibshirani, and J. H. Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. second ed. New York: Springer.

Hastie, T., R. Tibshirani, and M. Wainwright. 2015. *Statistical Learning with Sparsity: The Lasso and Generalizations*. New York: CRC Press.

Hoeffding, W. 1948. “A Class of Statistics with Asymptotically Normal Distribution”. *Annals of Mathematical Statistics* 19(3):293–325.

Hofert, M. and Lemieux, C. 2023. “CRAN: Package qrng”. https://cran.r-project.org/web/packages/qrng/index.html, accessed 13th August 2025.

Jiang, T. 2003. *Data-Driven Shrinkage Strategies for Quasi-Regression*. Ph.D. thesis, Department of Statistics, Stanford, USA. https://www.proquest.com/dissertations-theses/data-driven-shrinkage-strategies-quasi-regression/docview/305302653/se-2, accessed 13th August 2025.

Jiang, T., and A. B. Owen. 2002. “Quasi-Regression for Visualization and Interpretation of Black Box Functions”. Technical report, Stanford University, USA, https://artowen.su.domains/reports/qregvis.pdf, accessed 14th August 2025.

Jiang, T., and A. B. Owen. 2003. “Quasi-Regression with Shrinkage”. *Mathematics and Computers in Simulation* 62(3-6):231–241.

L’Ecuyer, P. 2018. “Randomized Quasi-Monte Carlo: An Introduction for Practitioners”. In *Monte Carlo and Quasi-Monte Carlo Methods*, edited by A. B. Owen and P. W. Glynn, 29–52. Cham: Springer.

L’Ecuyer, P., and C. Lemieux. 2002. “Recent Advances in Randomized Quasi-Monte Carlo Methods”. In *Modeling Uncertainty: An Examination of Stochastic Theory, Methods, and Applications*, edited by M. Dror, P. L’Ecuyer, and F. Szidarovszky, 419–474. New York: Springer.

Lemieux, C., and A. B. Owen. 2002. “Quasi-Regression and the Relative Importance of the ANOVA Components of a Function”. In *Monte Carlo and Quasi-Monte Carlo Methods 2000*, edited by K.-T. Fang, H. Niederreiter, and F. J. Hickernell, 331–344. Berlin: Springer.

Niederreiter, H. 1992. *Random Number Generation and Quasi-Monte Carlo Methods*, Volume 63 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Philadelphia, PA: SIAM.

Nuyens, D., and B. J. Waterhouse. 2012. “A Global Adaptive Quasi-Monte Carlo Algorithm for Functions of Low Truncation Dimension Applied to Problems from Finance”. In *Monte Carlo and Quasi-Monte Carlo Methods 2010*, edited by L. Plaskota and H. Woźniakowski, 589–607. Berlin: Springer.

Owen, A. B. 1997a. “Monte Carlo Variance of Scrambled Net Quadrature”. *SIAM Journal on Numerical Analysis* 34(5):1884–1910.

Owen, A. B. 1997b. “Scrambled Net Variance for Integrals of Smooth Functions”. *The Annals of Statistics* 25(4):1541–1562.

Owen, A. B. 2000. “Assessing Linearity in High Dimensions”. *The Annals of Statistics* 28(1):1–19.

Owen, A. B. 2021. “About the R function: rsobol”. https://artowen.su.domains/code/note.pdf, accessed 13th August 2025.

Ribeiro, M. T., S. Singh, and C. Guestrin. 2016. “"Why Should I Trust You?": Explaining the Predictions of Any Classifier”. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’16. August 13th-16th, San Francisco, USA, 1135-1144.

Schreiber, J., P. Batlle, D. Wicaksono, and M. Hecht. 2024. “PMBO: Enhancing Black-Box Optimization through Multivariate Polynomial Surrogates”. *arXiv preprint arXiv: 2403.07485*.

Sobol’, I. M. 1967. “On the Distribution of Points in a Cube and the Approximate Evaluation of Integrals”. *USSR Computational Mathematics and Mathematical Physics* 7(4):86–112.

Sobol’, I. M. 1990. “On Sensitivity Estimation for Nonlinear Mathematical Models”. *Matematicheskoe Modelirovanie* 2(1):112–118.

Sobol, I. M. 2001. “Global Sensitivity Indices for Nonlinear Mathematical Models and Their Monte Carlo Estimates”. *Mathematics and Computers in Simulation* 55(1-3):271–280.

Street, W. N., W. H. Wolberg, and O. L. Mangasarian. 1993. “Nuclear feature extraction for breast tumor diagnosis”. In *Biomedical Image Processing and Biomedical Visualization*, edited by R. S. Acharya and D. B. Goldgof, 861 – 870. Bellingham: Society of Photo-Optical Instrumentation Engineers.

Venables, W. N., and B. D. Ripley. 2013. *Modern Applied Statistics with S-PLUS*. New York: Springer.

Williams, C. K., and C. E. Rasmussen. 2006. *Gaussian Processes for Machine Learning*. Cambridge: MIT Press.

Živanović, R. 2021. “Sparse Grid Regression for Interpretation of Black Box Functions”. In *2021 29th Mediterranean Conference on Control and Automation (MED)*. June 22nd-25th, Puglia, Italy, 765-769.

AUTHOR BIOGRAPHIES

AMBROSE EMMETT-IWANIW is a PhD student in Statistics from the Department of Statistics and Actuarial Science at the University of Waterloo. His research interests include quasi-Monte Carlo applications, probabilistic machine learning, and time series analysis. Email address: arsemmet@uwaterloo.ca.

CHRISTIANE LEMIEUX is a professor in the Department of Statistics and Actuarial Science at the University of Waterloo. She obtained a PhD in Computer Science from the Université de Montréal in 2000. Her research interests include quasi-Monte Carlo constructions and applications, and dependence concepts in sampling. Email address: clemieux@uwaterloo.ca.