Université de Montréal

Randomized Quasi-Monte Carlo Methods for Density Estimation and Simulation of Markov Chains

par

Amal Ben Abdellah

Département d'informatique et de recherche opérationnelle

Faculté des arts et des sciences

Thèse présentée à la Faculté des études supérieures et postdoctorales

en vue de l'obtention du grade de

Philosophiæ Doctor (Ph.D.)

en Informatique

Février 2021

Université de Montréal

Faculté des études supérieures et postdoctorales

Cette thèse intitulée

Randomized Quasi-Monte Carlo Methods for Density Estimation and Simulation of Markov Chains

présentée par

Amal Ben Abdellah

a été évaluée par un jury composé des personnes suivantes :

Simon Lacoste-Julien
(président-rapporteur)

Pierre L'Ecuyer
(directeur de recherche)

Manuel Morales
(membre du jury)

Alexander Kreinin
(examinateur externe)

François Perron
(représentant du doyen de la FESP)

Summary

The Randomized Quasi Monte Carlo method (RQMC) is often used to estimate an integral over the s-dimensional unit cube $(0,1)^s$. This integral is interpreted as the mathematical expectation of some random variable X. It is well known that RQMC estimators can, under some conditions, converge at a faster rate than crude Monte Carlo estimators of the integral.

For Markov chains simulation on a large number of steps by using RQMC, little exists. The most promising approach proposed to date is the array-RQMC method. This method simulates n copies of the chain in parallel using a set of independent RQMC points at each step, and sorts the chains using a specific sorting function after each step. This method has given empirically significant results in terms of convergence rates on a few examples (i.e. a much better convergence rate than that observed with Monte Carlo standard). However, the convergence rates observed empirically have not yet been theoretically proven.

In the first part of this thesis, we examine how RQMC can improve the convergence rate when estimating not only X's expectation, but also its distribution. In the second part, we examine how RQMC can be used for Markov chains simulation on a large number of steps using the array-RQMC method.

Our thesis contains four articles. In the first article, we study the effectiveness of replacing Monte Carlo (MC) by either randomized quasi Monte Carlo (RQMC) or stratification to show how they can be applied to make samples more representative. Furthermore, we show how these methods can help to reduce the integrated variance (IV) and the mean integrated square error (MISE) for the kernel density estimators (KDEs). We provide both theoretical and empirical results on the convergence rates and show that the RQMC and stratified sampling estimators can achieve significant IV and MISE reductions with even faster convergence rates compared to MC in some situations, while leaving the bias unchanged. In the second article, we examine the combination of RQMC with a conditional Monte Carlo approach to density

estimation. This approach is defined by taking the stochastic derivative of a conditional CDF of X and provides a large improvement when applied.

Using array-RQMC in order to price an Asian option under an ordinary geometric Brownian motion process with fixed volatility has already been attempted in the past and a convergence rate of $\mathcal{O}(n^{-2})$ was observed for the variance. In the third article, we study the pricing of Asian options when the underlying process has stochastic volatility. More specifically, we examine the variance-gamma, Heston, and Ornstein-Uhlenbeck stochastic volatility models. We show how applying the array-RQMC method for pricing Asian and European options can significantly reduce the variance.

An efficient sample path algorithm called (fixed-step) τ-leaping can be used to simulate stochastic biological systems as well as well-stirred chemical reaction systems. The crude Monte Carlo (MC) method is a feasible approach when it comes to simulating these sample paths. Simulating the Markov chain for fixed-step τ-leaping via ordinary randomized quasi-Monte Carlo (RQMC) has already been explored empirically and, when the dimension of the problem increased, the convergence rate of the variance was realigned with those observed in several numerical experiments using MC. In the last article, we study the combination of array-RQMC with this algorithm and empirically demonstrate that array-RQMC provides a significant reduction in the variance compared to the standard MC algorithm.

Key words: simulation, quasi-Monte Carlo, Markov chain, variance reduction, density estimation.

Résumé

La méthode Quasi-Monte Carlo Randomisé (RQMC) est souvent utilisée pour estimer une intégrale sur le cube unitaire $(0,1)^s$ de dimension s. Cette intégrale est interprétée comme l'espérance mathématique d'une variable aléatoire X. Il est bien connu que, sous certaines conditions, les estimateurs d'intégrales par RQMC peuvent converger plus rapidement que les estimateurs par Monte Carlo.

Pour la simulation de chaînes de Markov sur un grand nombre d'étapes en utilisant RQMC, il existe peu de résultats. L'approche la plus prometteuse proposée à ce jour est la méthode array-RQMC. Cette méthode simule, en parallèle, n copies de la chaîne en utilisant un ensemble de points RQMC aléatoires et indépendants à chaque étape et trie ces chaînes en utilisant une fonction de tri spécifique après chaque étape. Cette méthode a donné, de manière empirique, des résultats significatifs sur quelques exemples (soit, un taux de convergence bien meilleur que celui observé avec Monte Carlo standard). Par contre, les taux de convergence observés empiriquement n'ont pas encore été prouvés théoriquement.

Dans la première partie de cette thèse, nous examinons comment RQMC peut améliorer, non seulement, le taux de convergence lors de l'estimation de l'espérance de X mais aussi lors de l'estimation de sa densité. Dans la deuxième partie, nous examinons comment RQMC peut être utilisé pour la simulation de chaînes de Markov sur un grand nombre d'étapes à l'aide de la méthode array-RQMC.

Notre thèse contient quatre articles. Dans le premier article, nous étudions l'efficacité gagnée en remplaçant Monte Carlo (MC) par les méthodes de Quasi-Monte Carlo Randomisé (RQMC) ainsi que celle de la stratification. Nous allons ensuite montrer comment ces méthodes peuvent être utilisées pour rendre un échantillon plus représentatif. De plus, nous allons montrer comment ces méthodes peuvent aider à réduire la variance intégrée (IV) et l'erreur quadratique moyenne intégrée (MISE) pour les estimateurs de densité par noyau (KDE).

Nous fournissons des résultats théoriques et empiriques sur les taux de convergence et nous montrons que les estimateurs par RQMC et par stratification peuvent atteindre des réductions significatives en IV et MISE ainsi que des taux de convergence encore plus rapides que MC pour certaines situations, tout en laissant le biais inchangé. Dans le deuxième article, nous examinons la combinaison de RQMC avec une approche Monte Carlo conditionnelle pour l'estimation de la densité. Cette approche est définie en prenant la dérivée stochastique d'une CDF conditionnelle de X et offre une grande amélioration lorsqu'elle est appliquée.

L'utilisation de la méthode array-RQMC pour évaluer une option asiatique sous un processus ordinaire de mouvement brownien géométrique avec une volatilité fixe a déjà été tentée dans le passé et un taux de convergence de $\mathcal{O}(n^{-2})$ a été observé pour la variance. Dans le troisième article, nous étudions le prix des options asiatiques lorsque le processus sous-jacent présente une volatilité stochastique. Plus spécifiquement, nous examinons les modèles de volatilité stochastique variance-gamma, Heston ainsi que Ornstein-Uhlenbeck. Nous montrons comment l'application de la méthode array-RQMC pour la détermination du prix des options asiatiques et européennes peut réduire considérablement la variance.

L'algorithme τ-leaping est utilisé dans la simulation des systèmes biologiques stochastiques. La méthode Monte Carlo (MC) est une approche possible pour la simulation de ces systèmes. Simuler la chaîne de Markov pour une discrétisation du temps de longueur τ via la méthode quasi-Monte Carlo randomisé (RQMC) a déjà été explorée empiriquement dans plusieurs expériences numériques et les taux de convergence observés pour la variance, lorsque la dimension augmente, s'alignent avec ceux observés avec MC. Dans le dernier article, nous étudions la combinaison de array-RQMC avec cet algorithme et démontrons empiriquement que array-RQMC fournit une réduction significative de la variance par rapport à la méthode de MC standard.

Mots-clés: simulation, quasi-Monte Carlo, chaînes de Markov, réduction de variance, estimation de la densité.

Contents

List of Tables

List of Figures

List of Abbreviations

AIV	Asymptotic Integrated Variance
AISB	Asymptotic Integrated Square Bias
AMISE	Asymptotic Mean Integrated Square Error
DTMC	Discrete Time Markov Chain
GMVT	Generalized Mean Value Theorem
GSS	Gamma Sequential Sampling
i.i.d	Independent and identically distributed
IV	Integrated Variance
ISB	Integrated Square Bias
KDE	Kernel Density Estimator
MC	Monte Carlo
MISE	Mean Integrated Square Error
NUS	Nested Uniform Scramble
QMC	Quasi Monte Carlo
RQMC	Randomized Quasi Monte Carlo
Var	Variance
VG	Variance Gamma
VRF	Variance Reduction Factor

Acknowledgements

I would like to express my sincere gratitude to my advisor Prof. Pierre L'Ecuyer for the continuous support of my Ph.D study and research, for his patience, motivation, enthusiasm, immense knowledge and for supporting me financially during my studies. His guidance helped me all along research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study.

This work has been supported by an IVADO Research Grant, an NSERC-Canada Discovery Grant, a Canada Research Chair, and an Inria International Chair, to Pierre L'Ecuyer.

I would also like to acknowledge Prof. Art Owen of the Department of Statistics at Stanford University as the co-author of one of my article, and I am gratefully indebted to his very valuable comments. Moreover, I would like to specially thank Florian Puchhammer, with who I collaborated a lot during this thesis, for his help and support.

Last but not least, I would like to thank my parents, for giving birth to me in the first place and supporting me spiritually throughout my life. I would like also to thank my sister, brothers, friends and all my Department colleagues for their support and encouragement.

Chapter 1

Introduction

1.1. MC, QMC, RQMC, and Array-RQMC

The Monte Carlo (MC) and quasi-Monte Carlo (QMC) methods are powerful tools for estimating high-dimensional integrals that can represent the mathematical expectation $\mathbb{E}[X]$ of a random variable X (Asmussen and Glynn, 2007; Dick and Pillichshammer, 2010; Glasserman, 2004; L'Ecuyer, 2009, 2018). The MC method estimates these integrals by using random sampling and it has a convergence rate of $\mathcal{O}(n^{-1/2})$ for the error, where n is the number of function evaluations. This convergence rate, though independent of the dimension, is very slow. The QMC methods replace random numbers with low discrepancy points or quasi-random numbers, which are deterministic. A QMC algorithm has a deterministic error bound of $\mathcal{O}(n^{-1}(\log\ n)^s)$ for functions of bounded variation in the sense of *Hardy-Krause (HK) variation*, where s is the dimension (Dick and Pillichshammer, 2010). This is asymptotically superior to the rate of MC. However, this method has difficulty producing a reliable error estimator.

The Randomized Quasi-Monte Carlo (RQMC) methods turn QMC into variance reduction methods by carefully randomizing the points set (L'Ecuyer, 2009, 2018; L'Ecuyer and Lemieux, 2000). They are hybrid techniques that combine clever integration rules with some randomness in order to provide an unbiased estimator with a variance bound that converges at a faster rate than the MC variance, under certain conditions. The classical variance bound would be the square of the QMC worst-case bound. But there are also certain RQMC settings in which the variance converges faster, e.g., at rate $\mathcal{O}(n^{-\alpha+\epsilon})$ for any $\epsilon > 0$, where α can be larger than 2 (L'Ecuyer et al., 2020a). On the other hand, this variance bound,

based on the KH inequality, is often very loose and practically useless when the dimension is more than a half-dozen, unless the sample size is astronomically large. Nevertheless, in applications, the true RQMC variance is often much smaller than the true MC variance, even if the bound is larger than the MC variance.

Some examples of RQMC point sets are: randomly shifted lattice rules, scrambled digital nets, and digital nets with a random digital shift. Another sampling method called stratified sampling is also used for integral estimation where the unit cube is partitionned into n boxes of equal volume and one point is generated randomly and uniformly in each box.

The array-RQMC algorithm was introduced in L'Ecuyer et al. (2006, 2008) for the express purpose of using RQMC for Markov chain simulation on a large number of steps. This method is designed to work with an array of discrete time Markov chains (DTMC) simulated in parallel and is suitable for Markov chains that evolve for a large number of steps. The idea of this method is to simulate n chains in parallel and use a RQMC point set to advance all the chains by one step at a time, and sorts the chains with a specific sorting function after each step. This method leads to highly promising empirical results (the observed convergenre rates are much better than for Monte Carlo) on a few applications, such as computational finance for instance. See L'Ecuyer et al. (2008).

1.2. Objectives and Contributions

In this thesis, we explore the effectiveness of replacing the MC algorithm by the RQMC algorithm or the stratified sampling algorithm over the unit cube to estimate the density of a random variable X. The goal is to reduce the integrated variance (IV) and the mean integrated square error (MISE) for two density estimators, histogram and kernel density estimator (KDE). We also propose a conditional Monte Carlo method (CMC) to estimate the density function of the output of a simulation model. By conditioning on a random variable, the empirical distribution can be smoothed, and we can take a derivative to obtain the density estimator and then apply RQMC to further improve the convergence rate of the MISE. Furthermore, we examine the performance of the array-RQMC method for the simulation of a molecular biological systems by testing it on several examples. We demonstrate the potential of the array-RQMC method when paired with these applications which have higher dimensional states and a larger number of reaction types. Moreover, we study the application

of array-RQMC for option pricing when the underlying process has stochastic volatility. We examine in particular the variance-gamma model, the Heston model, and the Ornstein-Uhlenbeck model.

This thesis involves two parts: the first part studies density estimation by using RQMC, and the second part studies the use of RQMC for the simulation of Markov chains.

1.2.1. Density estimation by RQMC methods

We consider a setting in which we want to estimate the distribution of $X = g(U)$ where $U = (U_1, \ldots, U_s) \sim U(0,1)^s$ (uniform over the unit hypercube) and $g : (0,1)^s \to R$. We suppose that X has density f over R. We aim to estimate f over some bounded interval $[a,b]$, supposing that $g(u)$ can be computed easily for any $u \in (0,1)^s$.

To estimate the density f, there are several approaches whose asymptotic convergence rates differ. In this thesis we denote by $\hat{f}_n$ the density estimator with a sample of size n, and the quality of the estimator can be measured by the *mean integrated square error* (MISE), defined as

$$\text{MISE} = \mathbb{E} \int_a^b (\hat{f}_n(x) - f(x))^2 dx,$$

which we want to minimize. The MISE may be decomposed as the sum of the *integrated variance* (IV) and the *integrated square bias* (ISB):

$$\text{MISE} = \text{IV} + \text{ISB} = \int_a^b \mathbb{E}(\hat{f}_n(x) - \mathbb{E}[\hat{f}_n(x)])^2 \mathrm{d}x + \int_a^b (\mathbb{E}[\hat{f}_n(x)] - f(x))^2 \mathrm{d}x.$$

A first (simple) density estimator is a *histogram* with bin (rectangle) width h, for which the MISE converges to 0 as $\mathcal{O}(n^{-2/3})$ if h is chosen optimally as a function of n. A *kernel density estimators* (KDE) is defined by selecting a *kernel density* k, and a constant $h > 0$ called the *bandwidth* that serves as a horizontal kernel stretching factor (Marron and Wand, 1992; Scott, 2015; Wand and Jones, 1995). Given a sample $X_1, \ldots, X_n$, the KDE is defined by

$$f_n(x) = \frac{1}{nh} \sum_{i=1}^{n} k\Big(\frac{x - X_i}{h}\Big) \tag{1.2.1}$$

for all $x \in \mathbb{R}$. For a KDE with optimal h, the MISE converges as $\mathcal{O}(n^{-4/5})$, for various choices of kernel. However, the optimal h (or a very good h) is usually hard to find in practice. Although kernel density estimators have better asymptotic theoretical properties

than histograms, the latter are still widely used by practitioners, mainly because of their simplicity and ease of interpretation.

Our first objective is to examine how replacing the Monte Carlo (MC) by the stratification and RQMC algorithms can help enhance, by using a kind of non-parametric density estimation mechanism (kernel density estimator), the asymptotic convergence rates of the IV and MISE. Then, we evaluate the improvements obtained relative to the IV and MISE by using an appropriate finite sample size n.

Note that for the kernel desnity estimator, with MC the IV and ISB converge as $\mathcal{O}(1/(nh))$ and $\mathcal{O}(h^4)$, respectively. Only the rate of the integrated bias differs. Replacing MC by RQMC with the same n and h may change the variance, but should not change the bias, because, for any $x \in [a,b]$, $\hat{f}_n(x)$ is an average which has the same expectation with RQMC as with MC. Therefore, we focus mostly on the behavior of the IV and we study how RQMC can reduce it and improve its rate of convergence.

With RQMC, We investigate the following model for the IV as a function of n and h :

$$\text{IV} = Cn^{-\beta}h^{-\delta} + o(n^{-\beta}h^{-\delta}) \tag{1.2.2}$$

when $n^{\beta}h^{\delta} \to \infty$ and $h \to 0$, for some parameters $C > 0$, $\beta > 0$, and $\delta > 0$ which depend on the problem and the RQMC method. We can estimate the parameters C, β, and δ by using linear regression on the log of the IV. The IV value is estimated based on experiments with several values of n and h. In those experiments, we will use powers of two from 2^{14} to 2^{19} for n, and a few values of h for each n.

The model presented in (1.2.2) will generally not work for the entire range of values of n and h. The idea is to find parameter values that provide a good approximation in a region that is relevant for the application at hand, and more particularly in the region where h minimizes the MISE as a function of n. In general, we do not know this region a priori, so, in practice, we may have to (re)learn this region given some model parameter estimates and (re)learn the model parameters given an estimate of the relevant region, iteratively. For given estimates of the model parameters, the optimal bandwidth h under this model can be estimated in the same way as with MC, but using the new expression for the IV.

Given that a KDE, at any given point, is an average (like the estimator of an expectation) it would seem natural to use RQMC to estimate the density as well, and to use the same

types of techniques (based on Koksma-Hlawka (HK)) to obtain variance and MISE bounds. At first, we expected that the HK inequality would provide bounds on the IV of the KDE and that this would, in turn, imply a faster convergence for RQCM than for MC. From there, we expected that the MISE would also follow a faster convergence rate. Unfortunately, our theoretical results did not agree with these expectations. It turns out that the KH bounds are not as good as we had hoped. We found that the best upper bound on the IV that KH gave us was $\mathcal{O}(n^{-2+\epsilon}h^{-2s})$ for any $\epsilon > 0$, while the ISB remained $\mathcal{O}(h^4)$ as with MC. This provided a bound of $\mathcal{O}(n^{-4/(2+s)+\epsilon})$ on the MISE if we choose h to minimize this bound. For comparison, the MISE rate for the KDE under MC is $\mathcal{O}(n^{-4/5})$. The factor h^{-2s} in the IV bound comes from the increase of the Hardy-Krause variation of each summand as a function of the underlying uniforms when h decreases. This increases the variation of the function that appears in the bound, and this impact grows exponentially in s. In order to exploit the smaller power of n in the IV bound to reduce the MISE bound, one must simultaneously decrease the ISB. By taking a smaller h, one can achieve this, which in turn drastically increases the IV bound. As a result, the bound on the MISE converges at a slower rate than anticipated and offers a slower rate compared to that observed with MC when $s \geq 4$. For a special type of RQMC method, namely a digital net with a nested uniform scramble, we also prove that the IV and MISE rates are never worse than those of MC (Ben Abdellah et al., 2019a).

For the combination of KDE with a stratification of the unit hypercube into subcubes, we also prove a different set of bounds on the IV and MISE, under the assumption that g is monotone. We obtain bounds that converge as $\mathcal{O}(n^{-(s+1)/s}h^{-2})$ for the IV and $\mathcal{O}(n^{-(2/3)(s+1)/s})$ for the MISE. The latter beats the MC rate for all $s < 5$. These bounds are obtained by bounding the variance directly. They do not involve the KH inequality nor the notion of variation and do not contain a power of $1/h$ that increases with s. We show examples where the IV and the MISE with stratification behave just like the bounds. These results do not imply that stratification works better than RQMC and the fact that the KH-based bounds on the IV and the MISE are not good for $s \geq 4$ does not mean that RQMC does not reduce the IV and the MISE. These are only upper bounds, and the true MISE may converge at a faster rate than these KH upper bounds (Ben Abdellah et al., 2019a).

It will be convenient to study empirically how the KDE really behaves in terms of IV and MISE under RQMC and stratification in actual simulations. With these sampling methods, we also need a procedure to select a good bandwidth h for the KDE as it will generally vary from a good h with MC. Our goal is to empirically evaluate the improvements in the IV and MISE achieved for reasonable sample sizes n in actual simulations. To do that, we use a regression model in log scale to estimate the IV and the MISE as functions of h and n, and the optimal h as a function of n, find a way to estimate its parameters (the constant term for the integrated bias expression is particularly difficult to estimate), test its goodness of fit, and then compare the accurate MISE estimates with RQMC to those obtained with standard MC. This model is expected to give a reasonably accurate approximation only for (n,h) in the selected region of interest and not necessarily everywhere, and the coefficients that we estimate may differ from the asymptotic ones.

We find that, empirically, combining RQMC with KDE often brings a significant improvement to the IV and the MISE compared to MC. This improvement is even present if there are more than three dimensions (which is a case where the KH bound converges more slowly than the MC rate for the KDE). Even when the MISE rate is not improved, there can still be a significant gain in the constant and in the actual MISE value. Moreover, in all our experiments, RQMC is never worse than MC. Proving those gains in theorems would be very hard, hence the importance of having various types of numerical examples.

Our second objective is to construct a density estimators that can avoid the bias. We first construct a smooth CDF estimator through conditional Monte Carlo (CMC) which we call *conditional density estimator* (CDE). The sample derivative of this estimator is used to estimate the density. Furthermore, this estimator is unbiased under appropriate conditions and its variance is uniformly bounded by a constant divided by n, so its MISE is $\mathcal{O}(n^{-1})$. The concept of using CMC was mentioned by Asmussen and Glynn (2007), page 146, Example 4.3, and further studied in Asmussen (2018). They estimated the density of a sum of continuous random variables each of which has a known distribution from which we can sample exactly. Asmussen (2018) merely removes one term of the sum and takes the conditional distribution of the sum given the other terms to estimate the cdf, the density, the value at risk, and the conditional value at risk of the sum.

Here, we estimate the CDF and density of X in a more general setting where we hide more than just one random variable to do the conditioning. We then provide conditions under which the density estimator can be demonstrated to be unbiased. A main condition is that the conditional CDF must be a continuous function at point x in which the density is estimated. Sometimes this can be accomplished by hiding, not only one, but multiple variables. The variance of the density estimator can only rely heavily on which variable(s) we hide, i.e., on what we are conditioning for.

Once we have a smooth density estimators, we can use a second approach to further enhance the convergence rate, by replacing the independent uniform random numbers driving the simulation with *randomized quasi-Monte Carlo* (RQMC) points. Under some conditions, we demonstrate that by combining these two approaches, we can obtain a density estimator whose MISE converges at a faster rate than $\mathcal{O}(n^{-1})$, namely $\mathcal{O}(n^{-2+\epsilon})$ for any $\epsilon > 0$. We also observe this faster rate empirically on numerical examples (L'Ecuyer et al., 2020b).

Other Monte Carlo density estimators were proposed recently, based on the concept of estimating the cdf derivative using a likelihood ratio (LR) method. This general approach permits to estimate the derivative of the expectation of a random variable with respect to some parameter of the underlying distribution when this random variable is discontinuous (Glynn, 1987; L'Ecuyer, 1990). The approach proposed by Laub et al. (2019) combines a clever change of variable with the LR method to estimate the density of a sum of random variables as in Asmussen (2018), but in a setting where the random variables are dependent. Peng et al. (2018) suggested a generalized version of the LR gradient estimator (GLR) to estimate the derivative of an expectation with respect to a more general model parameter. Lei et al. (2018) then sketched out how GLR could be used to estimate a density. We compare our method with these GLR-based estimators in our numerical illustrations.

1.2.2. RQMC for simulation of Markov Chains

Markov chains are used frequently in various areas, including statistics, machine learning, queueing theory, option pricing, etc. Numerical methods are available for computing steady-state probabilities or average cost/reward for Markov chain models. In principle, they could be applied to compute performance measures for entire simulation models. But from a practical viewpoint, these methods usually break down when the dimension of the state

space becomes too large, because they then require excessive computing times and memory. With simulation, in contrast, it is not the case when the state space is high-dimensional.

One of the robust and efficient Monte Carlo algorithm that has been established in order to efficiently explore a high dimensional space is the sequential Monte Carlo sampler (Del Moral et al., 2006). It is a class of algorithms for recursively computing Monte Carlo approximations of a sequence of distributions. The initial motivation of SMC was the filtering of state space models. They have been primarily used as "particle filters" to solve optimal filtering problems and, due to the fact that they provide a simple way of approximating complex filtering distribution sequentially in time, they have a wide-spread use in various applications including tracking, computer vision and digital communication. In addition, they give an unbiased estimate of the normalizing constant of the posterior distribution which can be one quantity of interest in the inference problem to deal with.

L'Ecuyer et al. (2007, 2008) have introduced a new RQMC algorithm, called array-RQMC, specially designed for Markov chains which evolves over several steps. They provide examples showing that their method can sometimes dramatically reduce the variance (or average square error), by several orders of magnitude compared with standard Monte Carlo. The idea of this method is to simulate n realizations of a Markov chain in such a way that each chain evolves according to its exact probability law, but the chains are not independent of each other. The goal is to induce some form of negative dependence between the realizations so that at any given step j, the distance (or discrepancy) between the empirical distribution of the n states and the theoretical distribution of the state at step j converges at a faster rate as a function of n compared to if the realizations were independent. This can improve the simulation efficiency for Markov chains simulated over several hundred steps. The applications include queueing systems, option pricing in finance, reliability and risk assessment models, and many more.

L'Ecuyer (2018), have empirically observed in a one-dimensional example that the convergence rate of the variance and the error converge as $\mathcal{O}(n^{-3/2})$ for a discontinuous cost function, a much faster rate (about $\mathcal{O}(n^{-3.4})$) is obtained for the variance when the function cost is continuously differentiable, and still about $\mathcal{O}(n^{-5/2})$ when the function cost is continuous but not differentiable. In a two-dimensional example of Asian options with a

continuous but not differentiable cost function, they have observed a $\mathcal{O}(n^{-2})$ convergence rate for the variance.

Array-RQMC in combination with several splitting variants and importance sampling were examined and tested by L'Ecuyer et al. (2007). Stratification improves the multinomial resampling process and it is shown that the combination with RQMC methods reduces variance and improve the efficiency factors. For multilevel splitting, the array-RQMC is applied separately for each step to estimate the corresponding probability. At any time a chain reaches the next level, it will be placed in a special state and waits there until the step is over. At the end of the step, we split the chains that have reached the target level k and restart the array-RQMC algorithm from there.

Pricing an American-style option and computing an optimal exercise strategy is a subclass of Markov decision process problem problems. It belongs to a large class of stochastic optimal stopping problems for which the standard tools are stochastic control theory and dynamic programming recurrence equations. When the state space is high-dimensional and/or the random variables have too complicated distributions of probability, these standard tools are not practical because the equations are difficult to solve numerically (Glasserman, 2004). Dion and L'Ecuyer (2010) have combined RQMC with approximate dynamic programming and, by making comparisons in terms of the expected values of the returned policies, it was empirically observed that RQMC reduces both the variance of bias between the expected payoff and the option price and additional to that, it returns better policies. For RQMC and array-RQMC, Sobol' nets with a linear scrambling and a random digital shift have been used, and the findings achieved are very similar for randomly-shifted lattice rule followed by baker's transformation.

Gerber and Chopin (2015) have proposed a sequential quasi Monte Carlo (SQMC) method which is a class of algorithms obtained by introducing QMC point sets into SMC. It is related to, and may be seen as an extension of, the array-RQMC algorithm to particle filtering. Its error rate is $o(n^{-1/2})$, which is smaller than the Monte Carlo rate, and $o(n^{-1})$ is the convergence rate for the variance. The complexity of SQMC is $\mathcal{O}(n \log n)$, where n is the number of simulations at each iteration. They have shown that SQMC is amenable to the same extensions as standard SMC, such as unbiased likelihood evaluation. They have

also obtained several convergence results and provided numerical evidence that SQMC may significantly outperform SMC in practical scenarios.

The pricing of an Asian option combined with array-RQMC, to estimate the expectation of its payoff, when the underlying process evolves as a geometric Brownian motion (GBM) with fixed volatility, and the state is two-dimensional (it contains the current value of the GBM and its running average) and a single random number is needed at each step, so the required RQMC points are three-dimensional, was already studied by L'Ecuyer (2018); L'Ecuyer et al. (2009) and they have observed a $\mathcal{O}(n^{-2})$ convergence rate for the variance, in a range of reasonable values of n, compared with $\mathcal{O}(n^{-1})$ for independent random points (Monte Carlo). For $n = 2^{20}$ (about one million chains), the variance ratio between Monte Carlo and Array-RQMC was around 2 to 4 millions.

Our third objective is to investigate how well the array-RQMC method would perform when the underlying process is more involved, e.g., when it has stochastic volatility. This is relevant because stochastic volatility models are more realistic than the plain GBM model (Madan and Seneta, 1990; Madan et al., 1998). Success is not guaranteed because the dimension of the required RQMC points is larger.

We extend this empirical analysis to price this option under a variance gamma (VG) process. The VG process is a Brownian motion with a random time change that follows a stationary gamma process. In that case the RQMC points must be four-dimensional instead of three-dimensional because the state has two dimensions and we need two uniform random numbers at each step. Also, we examine the Heston and Ornstein volatility models which are some of the most popular stochastic volatility option pricing models. They are motivated by the widespread evidence that volatility is stochastic and that the distribution of risky asset returns has tail(s) thicker than that of a normal distribution. For these models to price asian options, the RQMC points must be five-dimensional instead of three-dimensional and we need two uniform random numbers at each step. On the other hand, European options require a four-dimensional RQMC points (Ben Abdellah et al., 2019b).

Stochastic modelling of molecular biological systems has become an increasingly important area of research in recent years. Since intrinsic and extrinsic noise sources may play a crucial role in such systems, introducing randomness to the models appears to be a more and more popular and reasonable approach. We are particularly interested in chemical reaction

systems contained inside a single fixed volume, at constant temperature, where we assume that the system of molecules is well-mixed. Currently, the Stochastic Simulation Algorithm (SSA), introduced by Gillespie (1977), is probably one of the best known procedures for performing stochastic simulations of such systems of chemical reactions. SSA, however, has the drawback that it simulates each and every reaction that takes place in the system sequentially. Thus, this approach requires a large amount of computation time if the number of molecules or reactions is large.

To speed up the simulation, one can move from simulating a continuous time Markov chain to the simulation of a discrete time Markov chain (DTMC). More precisely, Gillespie (2001) introduced the τ-leap method as an approximate simulation strategy. Using Poisson random numbers for the number of reactions of each type that fire in a certain time interval of length τ, it is possible to leap over many reaction events in one step. If, in addition, τ is not chosen too large, this procedure still approximates the exact stochastic simulation well.

For the simulation of DTMCs several, viable, approaches are known, with simulation by crude Monte Carlo (MC) being probably the most prominent one. In many fields of application it is known, however, that replacing the i.i.d. points used for MC by randomized quasi-Monte Carlo (RQMC) point sets can yield a significant improvement in terms of variance reduction. In a recent paper, Beentjes and Baker (2019) examine whether the benefits of RQMC emerge in such chemical reaction systems with τ-leaping as well. Based on numerical experiments implemented for three testing examples, they found that RQMC indeed leads to a significant reduction of the variance. However, when the dimension of the RQMC point set is large, even though the convergence rates of the variance in terms of the number of points n looked well for small n (approximately n^{-2}), this rate appeared to deteriorate towards the MC rate of, roughly, n^{-1} for larger n.

Our fourth objective is to combine array-RQMC with the fixed-step τ-leaping and to examine the performance of this method in this molecular biological context by testing it on several examples. Furthermore, we want to demonstrate the potential of this method when we use standard multivariate sorts for further applications which have higher dimensional states and a larger number of reaction types. The structure of these examples enables us to define and use very simple sorting algorithms that significantly improve performance compared to the other multivariate sorts. A general approach for enhancing standard multivariate sorts

is to find an importance function that maps the states to the real numbers and then sort them by their mapped values. This not only reduces the dimensionality of the problem, but it can also increase efficiency if it is easy to evaluate the importance function. Naturally, this function should maintain all the significant state information at which it is assessed, which is not always achievable. As evidence that this strategy can work for chemical reaction networks, we generate sample data with plain MC and use it to fit very simple models of such importance functions. The variance we observe with this approach, in our examples, is always comparable to that of the best multivariate sorts. Note that we are not studying the behavior of the bias induced by τ-leaping for well-mixed chemical reaction network. We are rather investigating the performance of array-RQMC and how much it can decrease the variance of such simulations (Puchhammer et al., 2020).

1.3. Thesis structure

In the following we present the outline of the thesis:

- In chapter 2, a background is presented. We start by describing the principle characteristics of standard Monte Carlo (MC) and some examples of variance reduction techniques are described, like stratification. Also, we describe the principle characteristics of the Quasi-Monte Carlo (QMC) and Randomized Quasi-Monte Carlo (RQMC) methods. We discuss the main classes of point set constructions: lattice rules and digital nets, and the types of randomizations. The Array-RQMC method is described in more details with a presentation of the general principle as well as some discussion regarding its advantages over traditional Monte-Carlo algorithms, and we give general convergence results for this method.

- Chapter 3 presents the first article "Density estimation by Randomized Quasi-Monte Carlo", which has been accepted for publication in SIAM/ASA Journal on Uncertainty Quantification.

- Chapter 4 presents the second article "Monte Carlo and Quasi-Monte Carlo Density Estimation via Conditioning", which is currently under revision in INFORMS Journal on Computing.

- Chapter 5 presents the third article "Array-RQMC for option pricing under stochastic volatility models", which has been published in the proceedings of the Winter Simulation Conference.
- Chapter 6, presents the fourth article "Variance Reduction with Array-RQMC for Tau-Leaping Simulation of Stochastic Biological and Chemical Reaction Networks", which has been submitted for publication in The Bulletin of Mathematical Biology Journal.
- In Chapter 7, we finish this thesis by providing conclusions and future research perspectives that have arisen from the results of this dissertation.

Finally, we include the following three appendices:

- Appendix A: Supplements to Chapter 3 including detailed estimation results.
- Appendix B: Supplements to Chapter 5 including detailed estimation results.
- Appendix C: Supplements to Chapter 6, including further examples.

Chapter 2

Background on MC, QMC and Randomized QMC

In this chapter, we give an overview of general Monte Carlo techniques and we offer a basic review of quasi-Monte Carlo methods (QMC), the randomizations that turn them into variance-reduction techniques, the integration error and variance bound obtained for each method and the main classes of point set constructions: lattice rules and digital nets. The Array-RQMC method is described to develop the idea of using the RQMC method for the simulation of Markov chains.

2.1. Classical Monte Carlo Method

In the context of numerical integration, the principal idea of the Monte Carlo method is to estimate an integral by random sampling. In general, this integral is usually with respect to a non-uniform density π over the real espace

$$\mu = \int_{\mathbb{R}^s} g(\mathbf{x})\pi(\mathbf{x})\mathrm{d}\mathbf{x} = E_\pi[g(\mathbf{X})]. \tag{2.1.1}$$

This method draw an i.i.d sample from the distribution of interest π which can then be used to to approximate the average of $g(X)$. The basic idea is that any probability measure, π, defined with respect to a measure space, $\mathbb{R}^s$, can be approximated using the following empirical measure:

$$\pi^n(d\mathbf{x}) = \frac{1}{n}\sum_{i=0}^{n-1} \delta_{\mathbf{X}_i}(\mathrm{d}\mathbf{x}) \tag{2.1.2}$$

where $\{\mathbf{X}_i\}_{i=0}^{n-1}$, is a sequence of n i.i.d samples of law π, and one assumes $\pi(\mathrm{d}\mathbf{x})$ admits a density denoted $\pi(\mathbf{x})$ with respect to the Lebesgue measure.

The facility of computing this approximation has led to wide-spread use of Monte Carlo techniques, specifically with respect to approximating difficult integrals. In what is known as "crude Monte Carlo Sampling", one can generate samples, $\mathbf{X}_0$,...,$\mathbf{X}_{n-1}$, from the density $\pi(\mathbf{x})$. Then these samples may be used to obtain an empirical average, which can be used as an approximation to the solution of the integral in (2.1.1)

$$\hat{\mu}_n = \frac{1}{n}\sum_{i=0}^{n-1} g(\mathbf{X}_i). \tag{2.1.3}$$

Then, by applying the Strong Law of Large Numbers, it can be seen that $\hat{\mu}$ converges almost surely to μ and has variance σ^2/n, for a suitable class of functions g. We have

$$\sigma^2 := \mathrm{Var}[g(\mathbf{X}_i)] = \int_{\mathbb{R}^s} g^2(\mathbf{x})\pi(\mathbf{x})\mathrm{d}\mathbf{x} - \mu^2 \tag{2.1.4}$$

then, $\hat{\mu}_n$ is an unbiased estimator of μ and we have

$$E[\hat{\mu}_n] = \mu, \qquad \mathrm{Var}[\hat{\mu}_n] = \frac{\sigma^2}{n} = \mathcal{O}(n^{-1}) \tag{2.1.5}$$

The empirical variance of $g(\mathbf{x})$ is computed by

$$S_n^2 = \frac{1}{n}\sum_{i=0}^{n-1}\left[g(\mathbf{X}_i) - \hat{\mu}_n\right]^2. \tag{2.1.6}$$

Using this approximation and apply the Central Limit Theorem we have the following convergence

$$\sqrt{n}\frac{\hat{\mu}_n - \mu}{S_n} \Rightarrow \mathcal{N}(0,1). \tag{2.1.7}$$

This convergence of $\hat{\mu}_n$ permits the computation of an asymptotically valid confidence interval for μ, and the error $|\hat{\mu}_n - \mu|$ converge to zero as $\mathcal{O}(n^{-\frac{1}{2}})$ in probability.

By using a change of variable $g(\mathbf{X}_i) = f(\mathbf{U}_i)$ for some function $f : (0,1)^s \longrightarrow \mathbb{R}$, this integral can be written as an integral over the s-dimensional unit hypercube $(0,1)^s = \{\mathbf{u} = (u_1,...,u_s) : 0 < u_j < 1 \text{ for all } j\}$, i.e.,

$$\begin{aligned}\mu &= \int_0^1 ..... \int_0^1 f(u_1,...,u_s)\mathrm{d}u_1....\mathrm{d}u_s = \int_{(0,1)^s} f(\mathbf{u})\mathrm{d}\mathbf{u} \\ &= E(f(\mathbf{U})),\end{aligned} \tag{2.1.8}$$

where $\mathbf{u}$ represents a point in $(0,1)^s$, and $\mathbf{U} \sim U(0,1)^s$ is a random point with the uniform distribution over the unit hypercube. One can then rewrite (2.1.3) as

$$\hat{\mu}_n = \frac{1}{n} \sum_{i=0}^{n-1} f(\mathbf{U}_i) \tag{2.1.9}$$

where $\mathbf{U}_0$,...,$\mathbf{U}_{n-1}$ are independent random vectors uniformly distributed over $(0,1)^s$.

2.2. Variance Reduction Methods

The purpose of variance reduction techniques is to increase accuracy in the estimated variable by a decreased sample standard deviation per sample, instead of larger number of samples.

2.2.1. Stratified Sampling

This technique consists to partition the sample space into m pieces and involves sampling from each piece separately (Haber, 1966; L'Ecuyer and Buist, 2008). Suppose we want to compute the expectation

$$\mu = E(f(\mathbf{X})) = \int_{\mathbb{R}^s} f(\mathbf{x})\pi(\mathbf{x})\mathrm{d}\mathbf{x} \tag{2.2.1}$$

where $\mathbf{X}$ is the $\mathbb{R}^s$ valued random variable with density $\pi(x)$. Let $(D_i,\ 1 \leq i \leq m)$ be a partition of $\mathbb{R}^s$. The expectation can be written as

$$\mu = E(f(\mathbf{X})) = \sum_{i=1}^{m} E(\mathbb{I}_{(\mathbf{X}\in D_i)} f(\mathbf{X})) = \sum_{i=1}^{m} E(f(\mathbf{X})|\mathbf{X} \in D_i)P(\mathbf{X} \in D_i) \tag{2.2.2}$$

where $\mathbb{I}$ is the indicator function and

$$E(f(\mathbf{X})|\mathbf{X} \in D_i) = \frac{E(\mathbb{I}_{(\mathbf{X}\in D_i)} f(\mathbf{X}))}{P(\mathbf{X} \in D_i)}. \tag{2.2.3}$$

Let, $\mathbf{X}_i$ be a random variable whose law is the law of $\mathbf{X}$ conditioned by $(\mathbf{X} \in D_i)$, with density

$$\frac{\mathbb{I}_{(\mathbf{X}\in D_i)}\pi(\mathbf{X})}{\int_{D_i} \pi(\mathbf{y})d\mathbf{y}}. \tag{2.2.4}$$

Then $E(f(\mathbf{X})|\mathbf{X} \in D_i)$ can be interpreted as $E(f(\mathbf{X}_i))$. By using the Monte Carlo method, when the numbers $p_i = P(\mathbf{X} \in D_i)$ can be explicitly computed, we can approximate each conditional expectation $E(f(\mathbf{X})|\mathbf{X} \in D_i) = \mu_i$ by

$$\tilde{\mu}_i = \frac{(f(\mathbf{X}_{i,1}) + ... + f(\mathbf{X}_{i,n_i}))}{n_i} \tag{2.2.5}$$

where $\mathbf{X}_{i,1}, ..., \mathbf{X}_{i,n}$ are independent copies of $\mathbf{X}_i$. We have an estimator $\tilde{\mu}$ of μ:

$$\tilde{\mu} = \sum_{i=1}^{m} p_i \tilde{\mu}_i \tag{2.2.6}$$

Consequently, the variance of $\tilde{\mu}$ given by

$$\sum_{i=1}^{m} p_i^2 \frac{\sigma_i^2}{n_i}, \tag{2.2.7}$$

where, σ_i^2 is the variance of $f(\mathbf{X}_i)$. For the total number of the simulations $\sum_{i=1}^{m} n_i = n$ and

$$n_i = n \frac{p_i \sigma_i}{\sum_{i=1}^{m} p_i \sigma_i} \tag{2.2.8}$$

the variance of $\tilde{\mu}$ is

$$\mathrm{Var}(\tilde{\mu}) = \frac{1}{n} \left(\sum_{i=1}^{m} p_i \sigma_i \right)^2. \tag{2.2.9}$$

One can write

$$\begin{aligned}
\mathrm{Var}(f(\mathbf{X})) &= E(f(\mathbf{X})^2) - E(f(\mathbf{X}))^2 \\
&= \sum_{i=1}^{m} p_i E(f^2(\mathbf{X}) | \mathbf{X} \in D_i) - \left(\sum_{i=1}^{m} p_i E(f(\mathbf{X}) | \mathbf{X} \in D_i) \right)^2 \\
&= \sum_{i=1}^{m} p_i \mathrm{Var}(f(\mathbf{X}) | \mathbf{X} \in D_i) + \sum_{i=1}^{m} p_i E(f(\mathbf{X}) | \mathbf{X} \in D_i)^2 - \left(\sum_{i=1}^{m} p_i E(f(\mathbf{X}) | \mathbf{X} \in D_i) \right)^2.
\end{aligned} \tag{2.2.10}$$

According to the convexity inequality for $\mathbf{X}^2$, we get $(\sum_{i=1}^{m} p_i \sigma_i)^2 \leq \sum_{i=1}^{m} p_i \sigma_i^2$ if $\sum_{i=1}^{m} p_i = 1$ and, indeed, the variance is smaller than one obtained without stratification:

$$\mathrm{Var}(f(\mathbf{X})) \geq \sum_{i=1}^{m} p_i \mathrm{Var}(f(\mathbf{X}) | \mathbf{X} \in D_i) \geq \left(\sum_{i=1}^{m} p_i \sigma_i \right)^2. \tag{2.2.11}$$

If we want to stratify over the unit hypercube $[0,1]^s$, let $\{D_1,...,D_m\}$ be a partition of $[0,1]^s$. For example we can partition the unit cube into m rectangular boxes of equal size. Let $\{\mathbf{V}_1,...,\mathbf{V}_m\}$ be independent random variables, with $\mathbf{V}_i$ uniformly distributed over D_i. Then

$$\tilde{\mu} = \sum_{i=1}^{m} f(V_i) \tag{2.2.12}$$

is an unbiased estimator of μ and for a regular f, one has $\mathrm{Var}(\tilde{\mu})$ smaller that the variance by using MC. This technique was used by L'Ecuyer (2018) in a one dimensional example,

and they have observed an improvement from $\mathcal{O}(n^{-1})$ to at least $\mathcal{O}(n^{-3/2})$ for the rate of the mean square discrepancy and at least a rate of $\mathcal{O}(n^{-2})$ for the variance, for five types of function cost in an application of array-RQMC. A convergence rate of $\mathcal{O}(n^{-3/2})$ for the variance was proved in L'Ecuyer et al. (2008) for the case where the sample is stratified and the state is one-dimensional.

2.2.2. Conditional Monte Carlo

Conditional Monte Carlo (CMC) is a variance reduction method which improves the efficiency of a given estimator by conditioning it to a particular collection of information $\mathcal{L}$. That is, suppose that we want to estimate X, The CMC estimator can be written as $X_e = E(X|\mathcal{L})$, This estimator is clearly unbiased and by the Rao-Blackwell Theorem it follows that

$$\text{Var}[X] = \mathbb{E}[\text{Var}[X|\mathcal{L}]] + \text{Var}[\mathbb{E}[X|\mathcal{L}]], \tag{2.2.13}$$

Thus, we have

$$\text{Var}(X_e) = \text{Var}(X) - \mathbb{E}[\text{Var}[X|\mathcal{L}]] \leq \text{Var}(X). \tag{2.2.14}$$

Therefore, this method always provides a variance reduction by choosing a $\mathcal{L}$ such that it is possible to simulate the events in $\mathcal{L}$ and the conditional expectation $\mathbb{E}[X|\mathcal{L}]$ is known in closed form.

2.3. Quasi-Monte Carlo

QMC replaces the independent and identically distributed uniform random points $\mathbf{U}_i$ in (2.1.9) by a set of n points, $P_n = \{\mathbf{u}_0,...,\mathbf{u}_{n-1}\}$, chosen deterministically to be more uniform over the unit hypercube $(0,1)^s$ than a typical set of random points. The estimator $\hat{\mu}_n$ is replaced by the deterministic approximation

$$\bar{\mu}_n = \frac{1}{n}\sum_{i=0}^{n-1} f(u_i) \tag{2.3.1}$$

2.3.1. Discrepancy

There are many ways of measuring the uniformity of P_n, this is usually done via measures of non-uniformity called discrepancy. Formally, the general formula of discrepancy is defined

as

$$D(P_n;\mathbb{A}) = \sup_{A\in\mathbb{A}} \left| \frac{1}{n}\sum_{i=0}^{n-1} \mathbb{1}(\mathbf{u}_i \in A) - \lambda_s(A) \right| \tag{2.3.2}$$

where $\lambda_s(A)$ is the volume (Lebesgue measure on $\mathbb{R}^s$) of A, and $\mathbb{A}$ is a set of measurable sets. Two discrepancies are particularly useful: the extreme discrepancy, which is the discrepancy relative to the set $\mathbb{A}$ of s-dimensional intervals $\mathbf{[a,b]} := \Pi_{i=1}^{s}[a_i,b_i]$, $0 \le a_i < b_i < 1$,

$$D(P_n) = \sup_{\mathbf{[a,b]}} \left| \frac{1}{n}\sum_{i=0}^{n-1} \mathbb{1}(\mathbf{u}_i \in \mathbf{[a,b]}) - \Pi_{i=1}^{s}(b_i - a_i) \right| \tag{2.3.3}$$

and the star discrepancy, defined as

$$D^*(P_n) = \sup_{\mathbf{[0,b]}} \left| \frac{1}{n}\sum_{i=0}^{n-1} \mathbb{1}(\mathbf{u}_i \in \mathbf{[0,b]}) - \Pi_{i=1}^{s} b_i \right| \tag{2.3.4}$$

where again $\mathbf{[0,b]} := \Pi_{i=1}^{s}[0,b_i]$, $0 < b_i < 1$.

These two discrepancies are related as follows (Niederreiter, 1992)

$$D^*(P_n) \le D(P_n) \le 2^s D^*(P_n) \tag{2.3.5}$$

A general worst-case error bound is given by Koksma-Hlawka inequality

$$| \hat{\mu}_n - \mu | \le D^*(P_n) V(f) \tag{2.3.6}$$

where $V(f)$ can be interpreted as a measure of variation of the function f, and $D(P_n)$ is the discrepancy of P_n. Thus, if the function that we want to integrate has bounded variation $V(f)$, for a sequence of point sets $\{P_n, n \ge 1\}$, the integration error converges to zero at the worst at the same rate as $D(P_n)$. If this rate beats $\mathcal{O}(n^{-\frac{1}{2}})$, at that point we are doing asymptotically better than MC in two ways: the convergence is faster and we have a worst-case bound instead of just a confidence interval.

The quantity $V(f)$ is usually too hard to calculate in practice, and the inequality of Koksma-Hlawka is primarily used to determine the rate of the asymptotic error through $D^*(P_n)$. When P_n is a low-discrepancy point set so that $D^*(P_n) = \mathcal{O}(n^{-1}(\ln n)^s)$ and the integration error $| \hat{\mu}_n - \mu |$ converges as $\mathcal{O}(n^{-1}(\ln n)^s)$. This type of upper bound indicates that the advantage of the QMC methods over MC rate $\mathcal{O}(n^{-1/2})$ will eventually be lost as the dimension s increases (because the ln is not negligible when s increases), or more precisely, it suggests that QMC methods will require a sample size n too large, for practical purposes, to improve upon MC when s is large.

2.3.2. Construction of Point Sets

The two primary classes of approaches for constructing the point sets are lattice rules and digital nets (see Owen (1998a), Niederreiter (1992), and L'Ecuyer (2009)). An integration lattice is a vector space of the form

$$L_s = \left\{ \mathbf{v} = \sum_{j=1}^{s} h_j \mathbf{v}_j \text{ such that each } h_j \in \mathbb{Z} \right\}, \tag{2.3.7}$$

where $v_1,...,v_s \in \mathbb{R}^s$ are linearly independent over $\mathbb{R}$ and where L_s contains $\mathbb{Z}^s$, the set of integer vectors. The QMC approximation of μ with $P_n = L_s \cap [0,1)^s$ is a lattice rule, and the dual lattice is defined as

$$L_s^* = \left\{ \mathbf{h} \in \mathbb{R}^s : \mathbf{h}^\mathbf{t}\mathbf{v} \in \mathbb{Z} \text{ for all } \mathbf{v} \in L_s \right\}. \tag{2.3.8}$$

The rank of L_s is the smallest r such that one can find a basis of the form $v_1,...,v_s,e_{r+1},...,e_s$, where e_j is the j-th unit vector in s dimensions. The point sets corresponding to the lattice rule with a rank of 1 can be written as

$$\begin{aligned} P_n &= \{\mathbf{v} = i\mathbf{v}_1 \bmod 1,\ i = 0,...,n-1\} \\ &= \{(i\ \mathbf{a}_1 \bmod n)/n,\ i = 0,...,n-1\}, \end{aligned} \tag{2.3.9}$$

where $\mathbf{a}_1 = (a_1,...,a_s)$ and $\mathbf{v}_1 = \mathbf{a}_1/n$. The set P_n is fully projected-regular if and only if $r = 1$ and $\gcd(a_j,n) = 1$ for each j.

Korobov rules are a special case of lattice rules that are easy to implement, as we now describing. For a given sample size n, the only parameter required to generate a point set P_n in s dimensions is an integer a relatively prime to n . We then get

$$P_n = \{\frac{i}{n}(1,a,a^2,...,a^{s-1}) \bmod 1,\ i = 0,...,n-1\} \tag{2.3.10}$$

where the modulo 1 is applied component-wise.

Another main class of construction methods of low-discrepancy point sets and sequences is the digital nets. Let $b \geq 2$ be an arbitrary integer, usually a prime number, called the base. To define a net of $n = b^k$ points in s dimensions, we select s generator matrices $C_1,...,C_s$, which are matrices whose elements are in $Z_b = \{0,..,b-1\}$. The matrix C_j determines coordinate j of all the points, for $j \geq 1$. To define the i-th point u_i, for $i = 0,...,b^k - 1$ write the digital expansion of i in base b and multiply the vector of its digits by C_j, modulo b, to

obtain the digits of the expansion of $u_{i,j}$, the j-th coordinate of u_i. That is, the point set thus obtained is a digital net in base b.

$$i = \sum_{l=0}^{k-1} a_{i,l} b^l, \tag{2.3.11}$$

$$\begin{pmatrix} u_{i,j,1} \\ u_{i,j,2} \\ \vdots \end{pmatrix} = C_j \begin{pmatrix} a_{i,0} \\ a_{i,1} \\ \vdots \\ a_{i,k-1} \end{pmatrix} \bmod b, \tag{2.3.12}$$

$$u_{i,j} = \sum_{l=1}^{\infty} u_{i,j,l} b^{-l}, \tag{2.3.13}$$

$$\mathbf{u}_i = (u_{i,1},...,u_{i,s}). \tag{2.3.14}$$

A Sobol net is a special case of the digital net where $b = 2$ and the generator matrices C_j are upper triangular binary matrices with 1's on the diagonal :

$$C_j = \begin{pmatrix} 1 & v_{j,1,2} & \cdots & v_{j,1,c} & \cdots \\ 0 & 1 & \cdots & v_{j,2,c} & \cdots \\ \vdots & 0 & \ddots & \vdots & \\ & \vdots & & 1 & \end{pmatrix}. \tag{2.3.15}$$

The bits in column c of C_j can be represented as an odd integer smaller than 2^c:

$$\begin{aligned} m_{j,1} &= 1, \\ m_{j,2} &= 2v_{j,1,2} + 1, \\ &\vdots \\ m_{j,c} &= 2^{c-1} v_{j,1,c} + ... + 2v_{j,c-1,c} + 1 = \sum_{l=1}^{c} 2^{c-1} v_{j,l,c}, \end{aligned} \tag{2.3.16}$$

where $v_{j,c,c} = 1$. Thus, the column c contains the c digits of the binary expansion of $m_{j,c}$, from the most to the least significant, followed by zeros. Sobol' calls the real numbers $v_{j,c} = 2^{-c} m_{j,c}$, for $c = 1, ..., k$ and $j = 1, ..., s$, the direction numbers. To determine the integers $m_{j,c}$, for each j, we first choose a primitive polynomial over $\mathbb{F}_2$, say

$$f_j(z) = z^{d_j} + a_{j,1} z^{d_j - 1} + ... + a_{j,d_j}, \tag{2.3.17}$$

of degree d_j, and choose the first d_j integers $m_{j,1}, ..., m_{j,d_j}$. The integers $m_{j,d_j}, m_{j,d_j+1}, ...$ are then determined by the recurrence

$$m_{j,c} = 2a_{j,1}m_{j,c-1} \oplus ... \oplus 2^{d_j-1}a_{j,d_j-1}m_{j,c-d_j+1} \oplus 2^{d^j}m_{j,c-d_j} \oplus m_{j,c-d_j} \tag{2.3.18}$$

for $c \geq d_j$, or equivalently,

$$v_{j,l,c} = a_{j,1}v_{j,l,c-1} \oplus ... \oplus a_{j,d_j-1}v_{j,l,c-d_j+1} \oplus v_{j,l,c-d_j} \oplus v_{j,l+d_j,c-d_j} \tag{2.3.19}$$

for $l \geq 1$, where $\oplus$ denotes the bit-wise exclusive-or operation.

2.4. Randomized Quasi-Monte Carlo

The difficulty of obtaining accurate error estimators with QMC can be addressed by switching to randomized QMC (RQMC) which turns QMC into a variance-reduction technique (L'Ecuyer, 2009, 2018; L'Ecuyer and Lemieux, 2000, 2002; Owen, 1998a). The concept is to randomize P_n, in order to induce negative dependence between the points $\mathbf{u}_i$ by generating them as follows :

a) each point $\mathbf{u}_i$ is distributed uniformly over $[0,1)^s$, and

b) the point set $P_n = \{\mathbf{u}_0,...,\mathbf{u}_{n-1}\}$ covers $[0,1)^s$ more evenly than a set of independent random points.

RQMC point sets generate unbiased integral estimators, and the variance of the estimators can be estimated from independent estimator replicates. Then, we estimate μ by $\hat{\mu}_{\text{rqmc},n}$ which is defined as :

$$\hat{\mu}_{\text{rqmc},n} = \frac{1}{n}\sum_{i=0}^{n-1} f(\mathbf{u}_i). \tag{2.4.1}$$

This $\hat{\mu}_{rqmc,n}$ is an unbiased estimator of μ and, by taking m independent randomizations of P_n, an unbiased estimator of $\text{Var}[\hat{\mu}_{rqmc,n}]$ is obtained and a confidence interval for μ can eventually be calculated.

The difficulty encountered with all RQMC methods is that the dimension s can be very large and thus this RQMC scheme typically becomes ineffective.

2.4.1. Randomizations

We give a detailed description of the different randomizations that we will use in our simulation.

Random shift modulo 1 : given a single uniform random variable $\mathbf{U}$, it shift the entire point set P_n randomly, by adding $\mathbf{U}$ modulo 1, for each coordinate (Cranley and Patterson, 1976; L'Ecuyer and Lemieux, 2000). The randomized set $\tilde{P}_n := \{\tilde{\mathbf{u}}_i \ ; \ i = 0,1,...,n-1\}$ is defined by

$$\tilde{\mathbf{u}}_i = (\mathbf{u}_i + \mathbf{U}) \bmod 1. \tag{2.4.2}$$

Each shifted point is evenly distributed over $[0,1)^s$ with this randomization. Hence, even if the dimension s is much larger than the number of points n and if many coordinates of a given $\mathbf{u}_i$ are equal, these coordinates become independent after the randomization. Therefore each point has the same distribution as in the MC method; the only difference is that the n points of the shifted lattice are not independent.

Digital shift in base b : if P_n is a digital net in base b, an analogue method of the previous one is to write a b-ary representation of a vector $\mathbf{U}$ and add it to each point of P_n using operations over $\mathbb{F}_b$ (Dick and Pillichshammer, 2010; L'Ecuyer, 2018; L'Ecuyer and Lemieux, 1999, 2002). More precisely, if $\mathbf{U} = (U_1,...,U_s)$ and

$$\mathbf{U}_j = \sum_{l=1}^{\infty} d_{j,l} b^{-l}, \quad u_{i,j} = \sum_{l=1}^{\infty} u_{i,j,l} b^{-l}, \tag{2.4.3}$$

we compute

$$(\mathbf{u}_i + \mathbf{U}) \bmod b = (\tilde{u}_{i,1},...,\tilde{u}_{i,s}), \tag{2.4.4}$$

where

$$\tilde{u}_{i,j} = \sum_{l=1}^{\infty} ((u_{i,j,l} + d_{j,l}) \bmod b) b^{-l}. \tag{2.4.5}$$

This randomization was suggested for point sets based on linear recurrences modulo 2. It is also used in an arbitrary base. It is best suited for digital nets in base b and its application maintains any projection's uniformity.

Random Linear Matrix Scrambling : This randomization was proposed by Matoušek (1998) and discussed by Hong and Hickernell (2003); Owen (2003). It is obtained by applying a random matrix scramble, the idea is to generate s lower-triangular matrices $L_1,...,L_s$ with index in $N^* \times N^*$ and elements chosen randomly, independently over $\{0,...,b-1\}$

The digits $\tilde{u}_{i,j,1}$,$\tilde{u}_{i,j,2}$,... of a randomized coordinate $\tilde{u}_{i,j}$ are then obtained as

$$\begin{pmatrix} \tilde{u}_{i,j,1} \\ \tilde{u}_{i,j,2} \\ \vdots \end{pmatrix} = L_j \begin{pmatrix} u_{i,j,1} \\ u_{i,j,2} \\ \vdots \end{pmatrix} + d_j, \tag{2.4.6}$$

where the s vectors d_1,...,d_s with index in N^* and entries independently and uniformly distributed over $\{0,...,b-1\}$. A digital shift applied afterwards ensures that every point obtained is uniformly distributed in $[0,1)^s$. This technique preserves the properties of equidistribution of the original set.

Nested uniform scramble : Let $a \in [0,1)$ have base b expansion $a = \sum_{k=1}^{\infty} a_k b^{-k}$, where $a_k \in \mathbb{Z}_b$. We apply random permutations to the digits a_k yielding $\mathbf{u}_k \in \mathbb{Z}_b$ and deliver $\mathbf{u} = \sum_{k=1}^{\infty} \mathbf{u}_k b^{-k}$. $\mathbf{u}$ are generated as follows : $\mathbf{u}_1 = \pi_\bullet(a_1)$ where $\pi_\bullet$ is a uniform random permutation. Then $\mathbf{u}_2 = \pi_{\bullet a_1}(a_2)$, $\mathbf{u}_3 = \pi_{\bullet a_1 a_2}(a_3)$ and $\mathbf{u}_{k+1} = \pi_{\bullet a_1 a_2 ... a_k}(a_{k+1})$ where all of these permutations are independent and uniform. Owen (1995, 1997) proved that if the function f is periodic and enough smooth, for digital nets in base b with bounded t and fixed s, the variance converges as $\mathcal{O}(n^{-3}(\log(\mathrm{n}))^{s-1})$, which is better than the random digital shift.

2.4.2. Baker's transformation

For a randomly-shifted lattice rule, a simple technique that often reduces the variance significantly is the baker transformation, which transforms each coordinate u to $2u$ if $u < 1/2$ and to $2(1-u)$ if $u \geq 1/2$, this transformation improves the convergence rate when the function is non-periodic. Hickernell (2002) has shown that this technique reduces the variance to $\mathcal{O}(n^{-4+\epsilon})$ for non-periodic smooth functions. On the other hand, it also increases the variation of the integrand, so it may increase the variance (moderately) in other cases.

2.4.3. Variance Decomposition and Effective Dimension

When the dimension s is large, filling up the unit hypercube $[0,1)^s$ very uniformly is practically impossible. However, in many practical settings, the s-dimensional function f can be well approximated by a sum of lower-dimensional function, that depend only on a small number of coordinates of $\mathbf{u}$. The idea is to write f as :

$$f(\mathbf{u}) = \sum_{\mathfrak{v} \subseteq \{1,...,s\}} f_{\mathfrak{v}}(\mathbf{u}),$$

where $f_{\mathfrak{v}}(\mathbf{u})$ depends only on the coordinates of $\mathbf{u}$ that belong to the set $\mathbf{u}$, and the variance σ^2 decomposes as $\sigma^2 = \sum_{\mathfrak{v}\subseteq\{1,\ldots,s\}} \sigma^2_{\mathfrak{v}}$ where $\sigma^2_{\mathfrak{v}} = \mathrm{Var}[f_{\mathfrak{v}}(\mathbf{U})]$ for $\mathbf{U}$ uniformly distributed over $[0,1)^s$. This is known as an ANOVA decomposition of f. It often occurs that only a small fraction of the $\sigma^2_{\mathfrak{v}}$'s contribute to almost all the variance σ^2. Then, to obtain a significant variance reduction, it is sufficient to construct P_n so that the projections of the point P_n on those important subsets $\mathfrak{v}$ are very uniform. This can be realized by giving more weights to these projections in the discrepancy criterion used to construct the points.

We say that f has effective dimension d in proportion ρ in the superposition sense (Owen, 1998b) if

$$\sum_{|\mathfrak{v}|\le d} \sigma^2_{\mathfrak{v}} \ge \rho\sigma^2$$

If ρ is close to 1, this means that f is well approximated by a sum of d-dimensional (or less) functions.

We say that f has effective dimension d in proportion ρ in the successive-dimensions sense (L'Ecuyer and Lemieux, 2000) if

$$\sum_{\mathfrak{v}\subseteq\{i,\ldots,i+d-1\},0\le i\le s-d} \sigma^2_{\mathfrak{v}} \ge \rho\sigma^2$$

There are cases where the first few random numbers of the simulation are much more important than the others. If

$$\sum_{\mathfrak{v}\subseteq\{1,\ldots,d\}} \sigma^2_{\mathfrak{v}} \ge \rho\sigma^2,$$

then f has effective dimension d in proportion ρ in the truncation sense (Caflisch et al., 1997).

2.5. The Array-RQMC method

2.5.1. Markov Chain Model

We want to simulate n realizations of a Markov chain in a way that each chain evolves according to its exact probability law, but the chains are not independent of each other. We want to induce some form of negative dependence between the realizations so that at any given step j, the distance (or discrepancy) between the empirical distribution of the n states

and the theoretical distribution of the state at step j converges at a faster rate as a function of n than if the realizations were independent. That is, faster than $\mathcal{O}(n^{-1/2})$. There are methods such as randomized quasi-Monte Carlo (RQMC), stratified sampling, etc., that can give such super-canonical rates when estimating an integral by averaging values of a function over a highly-uniform (or low-discrepancy) point set. The difference here is that we want the low-discrepancy of the empirical distribution of the n states to be preserved from one step to the next when the Markov chains evolve.

Context and problem :

In the context of simulating Markov chains by randomized quasi-Monte Carlo (RQMC), we have the following setting proposed by L'Ecuyer et al. (2008, 2009), based on an idea that originates from Lécot (1989) for deterministic systems. In this case, we have a discrete-time Markov chain $\{X_j, j \geq 0\}$ defined by a stochastic recurrence over a measurable state space χ:

$$X_0 = x_0, \qquad X_j = \varphi_j(X_{j-1}, \mathbf{U}_j), \quad j \geq 1, \tag{2.5.1}$$

where $\mathbf{U}_1, \mathbf{U}_2, \ldots$ are independent random vectors uniformly distributed over the d-dimensional unit cube $[0,1)^d$ (hence-forth denoted $\mathbf{U}_j \sim U[0,1)^d$), for some functions $\varphi_j : \chi \times (0,1)^d \to \chi$. We want to estimate the expected total cost $\mu = E[Y]$, where

$$Y = \sum_{j=1}^{\tau} g(X_j) \tag{2.5.2}$$

for some measurable cost functions $g : \chi \to R$ and fixed time horizon τ.

There is a state-dependent cost $Y_j = g(X_j)$ at each step j. We estimate $E[Y_j]$ by the average (unbiased estimator) based on the n realizations $\{X_{i,j},\ j \geq 0\}$, $i = 0, \ldots, n-1$:

$$\bar{Y}_{n,j} = \frac{1}{n} \sum_{i=0}^{n-1} Y_{i,j} = \frac{1}{n} \sum_{i=0}^{n-1} g(X_{i,j}). \tag{2.5.3}$$

Estimation by MC : For $i = 0, \ldots, n-1$, generate $X_{i,j} = \varphi_j(X_{i,j-1}, \mathbf{U}_{i,j})$, $j = 1, \ldots, \tau$, where the $\mathbf{U}_{i,j}$'s are i.i.d. $U(0,1)^d$. We estimate μ by

$$\hat{\mu}_n = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{\tau} g(X_{i,j}) = \frac{1}{n} \sum_{i=1}^{n} Y_i = \frac{1}{n} \sum_{i=0}^{n-1} \sum_{j=1}^{\tau} g(\varphi(X_{i,j-1}, \mathbf{U}_{i,j})). \tag{2.5.4}$$

Estimation by QMC : Let $s = \tau d$, we replace the n independent random points $\mathbf{V}_i = (U_{i,1},...,U_{i,s})$ used in MC by a point set $P_n = (\mathbf{V}_0,....,\mathbf{V}_{n-1})$ that is evenly distributed over $[0,1]^s$, in order to reduce the error compared with MC.

Estimation by RQMC : The points V_i are randomized so that :

a) each point $\mathbf{V}_i$ is distributed uniformly over $[0,1)^s$, and

b) the point set $P_n = \{\mathbf{V}_0,...,\mathbf{V}_{n-1}\}$ covers $[0,1)^s$ more evenly than a set of independent random points.

We generate one RQMC point for each sample path, we put $\mathbf{V}_i = (\mathbf{U}_{i,1},...,\mathbf{U}_{i,\tau}) \in (0,1)^s = (0,1)^{d\tau}$. By doing this, we estimate μ by $\hat{\mu}_{rqmc,n}$ which is defined as :

$$\hat{\mu}_{rqmc,n} = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{\tau} g(X_{i,j}) \tag{2.5.5}$$

2.5.2. Array-RQMC method

The array-RQMC method was proposed by L'Ecuyer et al. (2008). Either τ is a random stopping time with respect to the filtration $\mathfrak{F}\{(j,X_j), j \geq 0\}$, $E[\tau] < \infty$, or τ is fixed, this method operates in the same way. It replaces the RQMC point set in $[0,1)^s$ by τ RQMC point sets in $[0,1)^d$ and simulates an array of n chains in parallel.

We use a mapping function h: $\chi \longrightarrow [0,1)^l$ for some small integer $l \geq 1$ where $\tilde{X}_j = h(X_j)$ contains all the information in X_j that is appropriate for the probability law of the future evolution of the chain. At each step, the n copies of the chain X_j are sorted in some order based on the transformed $\tilde{X}_j$, and then matched to n RQMC points based on that order. For some function $\tilde{\varphi}_j$ and if $\tilde{X}_{j-1}$ has a density f_j over $[0,1)^l$, let μ_j be the expected cost at step j, where

$$\mu_j = E[g_j(X_j)] = E[g_j(\tilde{\varphi}_j(\tilde{X}_{j-1},\mathbf{U}_j))] = \int_{[0,1)^{c+l}} f_j(\mathbf{w})g_j(\tilde{\varphi}_j(\mathbf{w},\mathbf{u}))d\mathbf{w}d\mathbf{u}. \tag{2.5.6}$$

For each step j, let $X_{0,j},...,X_{n-1,j}$ be the n realisations of X_j and $\tilde{X}_{i,j} = h(X_{i,j})$. Given that $\tilde{X}_{i,j-1}$ is assumed uniform over $[0,1)^l$ and $U_{i,j} \sim U([0,1)^d)$, we construct a $(d+l)$-dimensional RQMC point set defined as $Q_n = \{(\tilde{X}_{i,j-1},U_{i,j}), 0 \leq i < n\}$ with global negative dependence across the chains in a way that the empirical distibution of $S_{n,j} = \{X_{0,j},...,X_{n-1,j}\}$ provides a better approximation of the theoretical distribution of X_j than for MC. This property will have to be preserved at the next step, for the points $\tilde{X}_{i,j}$,

so we can use induction to argue that it holds at all steps. We construct a low discrepancy point set $\tilde{Q}_n = \{(w_i, U_{i,j}), 0 \le i \le n-1\}$ in which $w_i \in [0,1)^l$ are fixed and $X_{i,j}$ is computed by permuting the states $X_{i,j-1}$, by using a permutation π_j, so that $X_{\pi j(i),j-1}$ is close to w_i for each i and Q_n is close to $\tilde{Q}_n$ after the permutation of the points. The expected cost μ_j is then estimated by

$$\begin{aligned}\hat{\mu}_{arqmc,j,n} &= \frac{1}{n}\sum_{i=0}^{n-1} g_j(X_{i,j}) \\ &= \frac{1}{n}\sum_{i=0}^{n-1} g_j(\tilde{\varphi}_j(\tilde{X}_{\pi j(i),j-1}, \mathbf{U}_{i,j}))) \approx \frac{1}{n}\sum_{i=0}^{n-1}(g_j \circ \tilde{\varphi}_j)(w_i, \mathbf{U}_{i,j}))),\end{aligned} \tag{2.5.7}$$

which can be seen as a semi-RQMC estimator. The first l coordinates of the points are (in general) not randomized; they are only used to match the points to the chains. As a special case, if $c = 1$, one can take $w_i = (i + 0.5)/n$ or i/n and the best match is obtained by just sorting the states by increasing order of $\tilde{X}_{i,j-1}$. In this case, this w_i can be only implicit (not stored explicitly) because the points do not need to be sorted at each step.

2.5.3. Array-RQMC algorithm

Algorithm 1 describes in general the array-RQMC method such as μ is estimated by an unbiased estimator $\hat{\mu}_{\text{arqmc},n}$ and the empirical variance of m independent realizations of $\hat{\mu}_{arqmc,n}$ provides an unbiased estimator of $\text{Var}[\hat{\mu}_{arqmc,n}]$.

Algorithm 1 Array-RQMC Algorithm

$X_{i,0} \leftarrow x_0$ for $i = 0,...,n-1$;
for $j = 1,2,...,\tau$ **do**
  Compute an appropriate permutation π_j of the n states, based on the $h(X_{i,j-1})$, to match them with the RQMC points;
  Randomized afresh $\{\mathbf{U}_{0,j},...,\mathbf{U}_{n-1,j}\}$ in $\tilde{Q}_n$;
  let $i \leftarrow 0$;
  while $i < n$ **do**
    $X_{i,j} = \varphi_j(X_{\pi j(i),j-1}, \mathbf{U}_{i,j})$;
    $i \leftarrow i + 1$;
  end while
  $\hat{\mu}_{\text{arqmc},j,n} = \frac{1}{n}\sum_{i=0}^{n-1} g_j(X_{i,j})$
end for
Estimate μ by $\hat{\mu}_{arqmc,n} = \sum_{j=0}^{\tau} \hat{\mu}_{\text{arqmc},j,n}$

2.5.4. Sorting Strategies

When $l > 1$, the states and the points are mapped by applying a multivariate sort to the states in [0,1)^l. A multivariate batch sort is defined as follows, it separates the objects in n_1 batches of size n/n_1 such that each object in a batch is smaller or equal to the objects in the next batches and the states are sorted by the first coordinate. Then, each batch is separated in n_2 batches by the same way but the states in this case are sorted by the second coordinate. And so on. The integers n_1,n_2,...,n_d are chosen such as the product $n_1 n_2 ... n_d$ is equal to n. A variant of the batch sort in which $n = 2^e$ and $n_1 = n_2 = ... = n_e = 2$, is called split sort, it sorts by first coordinate in two packets and sort each packet by second coordinate in two packets, and so on, recursively. In these two sorts, the state space does not have to be mapped to the unit hypercube [0,1)^l.

A widely used technique for ordering multivariate objects is a space-filling curve. It is defined as a continuous map $H : [0,1] \longrightarrow [0,1]^l$ that fills completely $[0,1]^l$ and its pseudo-inverse $h : [0,1]^l \longrightarrow [0,1]$ verifing for any $x \in [0,1]^l$, $H \circ h(x) = x$. For $l = 1$, the map H and its pseudo-inverse h are the identity mappings, i.e. $H(x) = h(x) = x$, $\forall x \in [0,1]$ is used.

A discrete variant of the curve is described as follows, in base $b \geq 2$, at iteration level $m \geq 1$, the inverse is a map $h_m : \{0,1,...,b^m - 1\}^l \to \{0,1,...,b^{ml} - 1\}$. To partition $[0,1)^l$ into b^{ml} subcubes of the same size, the integer coordinates of a point $x = (x_1,...,x_l) \in [0,1)^l$ are defined by $(i_1,...,i_l)$ where $i_k = \lfloor b^m x_k \rfloor$. These integer coordinates identify the subcube that contains the point x and h_m enumerates these subcubes from 0 to $b^{ml} - 1$. If no subcube contains more than one point, the points will be sorted separately. If some subcube contains more than one point, we can split it again into b^l smaller subcubes by dividing each of its edges into b equal parts.

A Hilbert curve uses base $b = 2$. It divides the cube into 2^{ml} subcubes of equal size and while any subcube contains more than one point, it partitions it in 2^l and it defines a way to enumerate the subcubes so that successive subcubes are always adjacent, then, it map the states the points as if the state has one dimension and it uses a $(1 + d)$-dimensionnal RQMC point ordered by the first coordinate to match the states and the d randomized coordinates are used to advance the chains. The Hilbert batch sort and Hilbert curve split are an alternative of batch and split sort, one of these sorts may be applied to the states, so

that the n states are mapped to the n subboxes that partition $\mathbb{R}^l$, and then enumerate those subboxes (and states) in the same order as the Hilbert curve.

When $\chi \subseteq \mathbb{R}^l$, Gerber and Chopin (2015), used this sort after mapping the state to the unit hypercube $[0,1)^l$ via a logistic transformation defined as $\psi(x) = (\psi_1(x_1),...,\psi_l(x_l))$ where

$$\psi_i(x_i) = \left[1 + \exp\left(- \frac{x_i - \underline{x}_i}{\bar{x}_i - \underline{x}_i}\right)\right]^{-1}, \quad i = 1,...,l. \tag{2.5.8}$$

The constants $\bar{x}_i$, $\underline{x}_i$ are $\bar{x}_i = \mu_i + 2\sigma_i$ and $\underline{x}_i = \mu_i - 2\sigma_i$, where μ_i and σ_i are respectively the mean and the variance of the prior distribution.

2.5.5. Convergence results and proofs

In this section, we recall the bounds on the worst-case error and the variance already derived by L'Ecuyer et al. (2008) for a special case where $l = d = 1$. For more details, see L'Ecuyer (2018). The star discrepancy Δ_j of the states at each step j is defined as

$$\Delta_j = \sup_{x\in\chi} |\hat{F}_j(x) - F_j(x)|. \tag{2.5.9}$$

where $\hat{F}_j$ is the empirical distribution of the state X_j and if

$$V(g_j) = \int_0^1 \left| dg_j(x) \right| \tag{2.5.10}$$

denotes the corresponding variation of g_j and the standard Koksma-Hlawka inequality becomes :

$$|\bar{Y}_{n,j} - \mu_j| \leq \Delta_j . V(g_j). \tag{2.5.11}$$

The square L_2 discrepancy at step j is

$$D_j^2 = \int_0^1 (\hat{F}_j(x) - F_j(x))^2 dx = \frac{1}{12n^2} + \frac{1}{n}\sum_{i=0}^{n-1} ((i + 0.5/n) - F_j(X_{(i),j}))^2 \tag{2.5.12}$$

where the $X_{(i),j}$ are the order statistic of $X_{0,j},...,X_{n-1,j}$. Then, $D_j^2 \leq \Delta_j^2$. With the corresponding square variation

$$V_2^2(g_j) = \int_0^1 \left(\frac{dg_j(x)}{dx} \right)^2 dx \tag{2.5.13}$$

We have the variance bound

$$\text{Var}[\bar{Y}_{n,j}] = \mathbb{E}[(\bar{Y}_{n,j} - \mathbb{E}[g_j(X_j)])^2] \leq \mathbb{E}[D_j^2 V_2^2(g_j)] \tag{2.5.14}$$

The goal is to bound the term Δ_j. The following assumptions are considered :

Assumption 1. *Suppose that l=d=1 and $\varphi_j(x,u)$ is a non-decreasing function in u, and by using the inversion method to simulate the next state from the cdf $F_j(z|.) = P[X_j \leq z|X_{j-1} = .]$.*

Define

$$\Lambda_j = \sup_{0 \leq z \leq 1} V(F_j(z|.)) \tag{2.5.15}$$

which is an upper bound on the variation of the X_j cdf as a function of the previous state.

Assumption 2. *Suppose that each square of the $k \times k$ (such as $n = k^2$) grid contains exactly one RQMC point.*

Proposition 3. *Under Assumptions 1 and 2,*

$$\Delta_j \leq n^{-1/2} \sum_{l=1}^{j} (\Lambda_l + 1) \prod_{i=l+1}^{j} \Lambda_i. \tag{2.5.16}$$

Corollary 4. *If $\Lambda_j \leq \rho < 1$ for all j, then*

$$\Delta_j \leq \frac{1+\rho}{1-\rho} n^{-1/2}. \tag{2.5.17}$$

Using this Corollary to bound D_j^2 can only give a bound of $\mathcal{O}(1/n)$ for the variance, which does not beat the MC rate.

2.5.5.1. *Variance bound for stratified sampling*

For $l = 1$, $d = 1$, and by using the stratified sample RQMC points, the unit cube is partitionned in $n = k^2$ small cubes, that there is exactly one RQMC point in each cube, and that the randomized parts of these points are independent and uniformly distributed in the cubes. The first coordinate of point i is deterministic and can be fixed to w_i and alternatively, the two coordinates of each point can be generated in its subcube, and the points can be sorted by their first coordinate.

Assumption 5. *Let Assumption 2 holds and suppose that the coordinates of the points are independent and uniformly distributed in the cube.*

Proposition 6. *Under Assumption 5, we have*

$$\mathbb{E}(D_j^2) \leq n^{-3/2}\left(\frac{1}{4}\sum_{l=1}^{j}(\Lambda_l + 1)\prod_{i=l+1}^{j}\Lambda_i^2\right) \tag{2.5.18}$$

Corollary 7. *If $\Lambda_j \leq \rho < 1$ for all j, then*

$$\mathbb{E}[D_j^2] \leq \frac{1+\rho}{4(1-\rho^2)}n^{-3/2} = \frac{1}{4(1-\rho)}n^{-3/2} \tag{2.5.19}$$

and therefore

$$\mathrm{Var}[\bar{Y}_{n,j}] \leq \frac{1}{4(1-\rho)}V_2^2(g_j)n^{-3/2} \tag{2.5.20}$$

Under these assumptions, it was proved that the variance converges as $\mathcal{O}(n^{-3/2})$ and the worst-case error converges as $\mathcal{O}(n^{-1/2})$.

This gives a slightly better rate than MC for all d, and this bound increases in j as $\mathcal{O}(j^2)$. For $l \geq 1$, the worst case error converges as $\mathcal{O}(n^{-1/(l+1)})$ for a discrete state space in $\chi \subseteq \mathbb{Z}^l$ (El Haddad et al., 2008), and in for a continuous state space $\chi \subseteq \mathbb{R}^l$, under some conditions on the function φ_j and by using a batch sort (El Haddad et al., 2010).

Chapter 3

Article 1: Density Estimation by Randomized Quasi-Monte Carlo

In the first article, we introduce a randomized quasi-Monte Carlo(RQMC) based method to estimate the density f of some real valued random variable $X = g(\mathbf{U})$ where $\mathbf{U} = (U_1, \ldots, U_s) \sim U[0,1]^s$ (uniform over the unit hypercube) and $g : [0,1]^s \to \mathbb{R}$ can be evaluated point-wise. The basic idea of the proposed approach is to first generate a sample $(X_1, \cdots, X_n) = (g(\mathbf{U}_1), \cdots, g(\mathbf{U}_n))$, where $\{\mathbf{U}_1, \ldots, \mathbf{U}_n\}$ is an RQMC point set, and then estimate f using a standard density estimator, such as the kernel density estimator or the histogram. For these two estimators a bandwidth h that allows to control the trade-off between bias (or more precisely the integrated square bias, ISB) and variance (or more precisely the integrated variance, IV) should be specified. The asymptotic behaviour of the mean integrated square error (MISE), defined as MISE=IV+ISB, is analysed and compared with the MISE obtained when a plain Monte Carlo (MC) scheme is used, that is when $\{\mathbf{U}_1, \ldots, \mathbf{U}_n\}$ is a set of n i.i.d. $U(0,1)^s$ random variables. For a given choice of bandwidth h, the bias term is identical with both RQMC and MC while we show that the variance term is smaller with RQMC than with MC. The interesting consequence of this result is that, for a given n, the optimal bandwidth h is smaller with RQMC than with MC. We provide rigorous proofs for the RQMC bounds in s dimensions, for both RQMC and stratification. We also have new sets of bounds showing that the MISE rate cannot be worse than for crude Monte Carlo for one form of RQMC and for stratification. The article ends up with a numerical study where the good performance of the RQMC based method is illustrated on various examples.

This article has been accepted for publication in SIAM/ASA Journal on Uncertainty Quantification. Preliminary work was presented at the following conferences:

— The Optimization Days, Montreal, May 2019;

— Meet a GERAD Researcher conference, GERAD, Montreal, March 2019;

— Workshop on Frontier Technologies for High-Dimensional Problems and Uncertainty Quantification, RICAM, Linz, Austria, Dec. 2018;

— MCQMC 2018 Conference, Rennes, France, July 2018;

— School of Mathematics and Statistics, University of New South Wales, Sydney, Australia, March 2018;

— MCM 2017: Eleventh International Conference on Monte Carlo Methods and Applications, Montreal, July 2017.

The main author contributions are:

— The general research ideas were proposed by Pierre L'Ecuyer and Art B. Owen;

— The theoretical results were developed by all four authors.

— The implementations and experiments were carried out by Amal Ben Abdellah with the help of Florian Puchhammer.

— The article was written jointly.

Density estimation by Randomized Quasi-Monte Carlo

Amal Ben Abdellah[1], Pierre L'Ecuyer[1], Art B. Owen[2] and Florian Puchhammer[1]

[1] DIRO, University of Montreal, 2920 Chemin de La Tour, Pavillon Aisenstadt, Montreal, QC, H3T 1N8, Canada.

[2] Department of Statistics, Stanford University, Sequoia Hall, 390 Serra Mall, Stanford, CA, 94305-4065, USA.

Funding: This work has been supported by a Canada Research Chair, an Inria International Chair, an IVADO Grant, and NSERC Discovery Grant number RGPIN-110050 to P. L'Ecuyer. A. B. Owen was supported by the US National Science Foundation under Grants DMS-1521145 and DMS-1407397.

Abstract

We consider the problem of estimating the density of a random variable X that can be sampled exactly by Monte Carlo (MC). We investigate the effectiveness of replacing MC by randomized quasi Monte Carlo (RQMC) or by stratified sampling over the unit cube, to reduce the integrated variance (IV) and the mean integrated square error (MISE) for kernel density estimators. We show theoretically and empirically that the RQMC and stratified estimators can achieve substantial reductions of the IV and the MISE, and even faster convergence rates than MC in some situations, while leaving the bias unchanged. We also show that the variance bounds obtained via a traditional Koksma-Hlawka-type inequality for RQMC are much too loose to be useful when the dimension of the problem exceeds a few units. We describe an alternative way to estimate the IV, a good bandwidth, and the MISE, under RQMC or stratification, and we show empirically that in some situations, the MISE can be reduced significantly even in high-dimensional settings. density estimation;

Key words: Density estimation, quasi-Monte Carlo, stratification, variance reduction, kernel density, simulation.

3.1. Introduction

We are interested in estimating by simulation the density of a random variable $X = g(\mathbf{U})$ where $\mathbf{U} = (U_1, \ldots, U_s) \sim U[0,1]^s$ (uniform over the unit hypercube) and $g : [0,1]^s \to \mathbb{R}$. We assume that $g(\mathbf{u})$ can be computed easily for any $\mathbf{u} \in [0,1]^s$, that X has density f (with

respect to the Lebesgue measure) over $\mathbb{R}$ and we want to estimate f over some bounded interval $[a,b]$. A flurry of stochastic simulation applications fit this framework; see Asmussen and Glynn (2007); Law (2014), for example. The vector $\mathbf{U}$ represents the independent uniform random numbers that drive the simulation.

We denote by $\hat{f}_n$ a density estimator based on a sample of size n, and we measure the quality of the estimator over $[a,b]$ by the *mean integrated square error* (MISE), defined as

$$\text{MISE} = \int_a^b \mathbb{E}[\hat{f}_n(x) - f(x)]^2 \mathrm{d}x,$$

which we want to minimize. The MISE can be decomposed as the sum of the *integrated variance* (IV) and the *integrated square bias* (ISB):

$$\text{MISE} = \text{IV} + \text{ISB} = \int_a^b \mathbb{E}(\hat{f}_n(x) - \mathbb{E}[\hat{f}_n(x)])^2 \mathrm{d}x + \int_a^b (\mathbb{E}[\hat{f}_n(x)] - f(x))^2 \mathrm{d}x.$$

Minimizing the MISE generally involves a bias-variance tradeoff.

The density is often estimated by a histogram for visualization, but one can do better with more refined techniques, such as a *kernel density estimator* (KDE), defined as follows. One selects a *kernel* $k : \mathbb{R} \to \mathbb{R}$, and a constant $h > 0$ called the *bandwidth*, which acts as a horizontal stretching factor for the kernel. The kernels considered here are smooth probability densities that are symmetric about 0. In our experiments, we will use the Gaussian kernel, which is the standard normal density. Given a sample $X_1, \ldots, X_n$, the KDE at $x \in \mathbb{R}$ is

$$\hat{f}_n(x) = \frac{1}{nh} \sum_{i=1}^{n} k\left(\frac{x - X_i}{h}\right). \tag{3.1.1}$$

Density estimation methods such as KDEs were developed for the context where an independent sample $X_1, \ldots, X_n$ from the unknown density f is given. Here we assume that we can generate a sample of arbitrary size by choosing where to sample. With *crude Monte Carlo* (MC), we would estimate the density from a sample $X_1, \ldots, X_n$ of n *independent* realizations of X, obtained by simulation. Then the analysis is the same as if the data was collected from the real world, and the standard KDE methodology would apply (Scott, 2015). In that context, the IV is $\mathcal{O}(1/nh)$ and the ISB is $\mathcal{O}(h^4)$, so the MISE is $\mathcal{O}(n^{-4/5})$ if h is chosen optimally. This is slower than the $\mathcal{O}(n^{-1})$ canonical rate for the variance when estimating the mean.

Our aim in this paper is to study if, when, and how using *randomized quasi-Monte Carlo* (RQMC) or *stratification* can provide a KDE with a smaller MISE than with crude MC. It is well known that when we estimate the mean $\mathbb{E}[X]$ by the average $\bar{X}_n = (X_1 + \cdots + X_n)/n$, under appropriate conditions, using RQMC provides an unbiased estimator whose variance converges at a faster rate (in n) than the MC variance (Dick and Pillichshammer, 2010; L'Ecuyer, 2018; L'Ecuyer and Lemieux, 2002; Owen, 1997, 1998b). This variance bound is easily proved by squaring a worst-case deterministic error bound obtained via a version of the *Koksma-Hlawka* (KH) inequality, which is a Hölder-type inequality that bounds the worst-case integration error by a product of the variation of g and the discrepancy of the set of points $\mathbf{U}$ at which g is evaluated. Hundreds of papers have studied this. Of course, the faster rate is an asymptotic property and the KH bound may hide a large constant factor, so it could happen that this bound is larger than the MC variance for a given n. But in applications, the true RQMC variance is often much smaller than both the bound and the MC variance, even for moderate sample sizes. The bottom line is that RQMC is practically useful in many applications, when estimating the mean by an average. Stratification of the unit hypercube also provably reduces the variance of $\bar{X}_n$, although its applicability degrades quickly with the dimension, and it is typically dominated by RQMC when the dimension exceeds 1 or 2 (L'Ecuyer, 2018).

Since the KDE (4.2.1) at any given point is an average just like the estimator of an expectation, it seems natural to use RQMC to estimate a density as well, and to derive variance bounds via the same methods as for the mean estimator. This was the starting point of this paper. At first, we thought that the KH inequality would provide bounds on the IV of the KDE that converge faster for RQMC than for MC, and that a faster convergence rate of the MISE would follow. But things are not so simple. The best upper bound on the IV that KH gave us is $\mathcal{O}(n^{-2+\epsilon}h^{-2s})$ for any $\epsilon > 0$, while the ISB remains $\mathcal{O}(h^4)$ as with MC. This gives a bound of $\mathcal{O}(n^{-4/(2+s)+\epsilon})$ on the MISE if we select h to minimize this bound. The unwelcome h^{-2s} factor in the IV bound comes from the increase of the Hardy-Krause variation of each summand in (4.2.1) as a function of the underlying uniforms when h decreases. This effect grows exponentially in s. To exploit the smaller power of n in the IV bound to reduce the MISE bound, one must simultaneously decrease the ISB. One can achieve this by taking a smaller h, which in turn drastically increases the IV bound. This

limits seriously the rate at which the MISE bound can converge. The resulting rate for the bound beats the MC rate only for $s < 3$. For a special type of RQMC method, namely a digital net with a nested uniform scramble, we also prove that the IV and MISE rates are never worse than for MC.

For the KDE combined with a stratification of the unit hypercube into subcubes, which could be seen as a weak form of RQMC, we obtain bounds that converge as $\mathcal{O}(n^{-(s+1)/s}h^{-2})$ for the IV and $\mathcal{O}(n^{-(2/3)(s+1)/s})$ for the MISE. The latter beats the MC rate for all $s < 5$. These bounds are proved using arguments that do not involve KH and they are tight. We show examples where the IV and the MISE with stratification behave just like the bounds.

These results do not imply that stratification works better than RQMC, or that RQMC does not beat MC in more than two dimensions. The KH bounds are *only upper bounds* and nothing precludes that the true IV and MISE can be significantly smaller with RQMC than with MC or stratification, even if the RQMC variance bound is larger and converges more slowly. At a minimum, we should test empirically how the KDE really behaves in terms of IV and MISE under RQMC and under stratification. We also need a procedure to choose a good bandwidth h for the KDE with these sampling methods, since it will generally differ from a good h with MC. We do that in the second half of the paper. Our aim is to assess empirically the improvements achieved for reasonable sample sizes n in actual simulations. We use a regression model in log scale to estimate the IV and the MISE as functions of h and n, and the optimal h as a function of n. We find that RQMC often reduces the IV and the MISE significantly, even in more than 3 dimensions, and that it performs better than stratification. Sometimes, the convergence rate of the MISE is not improved but there is a significant gain in the constant and in the actual MISE. In all our experiments, the MISE was never larger with RQMC or stratification than with MC. We prove that this always holds for stratification. But for RQMC, we think that proving the observed gains in theorems would be very hard, hence the importance of testing with diverse numerical examples.

The remainder is organized as follows. In Section 3.2, we recall the definitions and basic properties of KDEs, including a strategy to find a good h under MC. In Section 3.3, we recall classical error and variance bounds for RQMC integration. In Section 3.4, we use classical QMC theory to derive KH bounds on the IV and the MISE for a KDE under RQMC, under reasonable assumptions. In Section 3.5 we derive IV and MISE bounds for

a KDE combined with stratification. We have bounds that converge at a faster rate than for MC when the dimension is small. We also show that stratification never increases the IV or MISE compared with MC. In Section 3.6, we report on numerical experiments in which we estimate and compare the true IV and MISE of the KDE with MC, RQMC, and stratification, for various examples. We also provide a method to find a good bandwidth h, which is necessary for their effective implementation, and we use a regression model to capture how the IV and the MISE really behave in the examples. We give our conclusions in Section 5.6. A supplement available online contains additional numerical results, goodness-of-fit tests, etc. It also provides results for RQMC and stratification for histograms instead of KDEs.

We adopt the usual $\Theta(\cdot)$ notation for the *exact order*: $h(n) = \Theta(\varphi(n))$ means that there is some n_0 and constants $c_2 > c_1 > 0$ such that for all $n \geq n_0$, $c_1 \leq h(n)/\varphi(n) \leq c_2$. This is less restrictive than $h(n) \propto \varphi(n)$. Also, $c(n,h) = \mathcal{O}(\varphi(n,h))$ means that there is a constant $K > 0$ such that for all integers $n \geq 1$ and all $h \in (0,1]$, $c(n,h) \leq K\varphi(n,h)$.

3.2. Kernel density estimators with MC

We recall asymptotic properties of the KDE with MC when $nh \to \infty$ and $h \to 0$ together. The details can be found in Jones et al. (1996); Scott (2015); Wand and Jones (1995), for example. The asymptotic MISE, IV, and ISB in this regime are denoted AMISE, AIV, and AISB, respectively. If $\mathrm{IV}(n,h)$ denotes the IV for a given (n,h), writing $\mathrm{AIV} = \tilde{g}(n,h)$ for some function $\tilde{g}$ means that $\lim_{n\to\infty,\, \tilde{g}(n,h)\to 0} \mathrm{IV}(n,h)/\tilde{g}(n,h) = 1$ and similarly for the AMISE and AISB. For measurable functions $\psi : \mathbb{R} \to \mathbb{R}$, we define the roughness functional $R(\psi) = \int_a^b (\psi(x))^2 \mathrm{d}x$ and the "moments" $\mu_r(\psi) = \int_{-\infty}^{\infty} x^r \psi(x) \mathrm{d}x$, for integers $r \geq 0$. We make the following assumptions in the rest of the paper.

Assumption 3.2.1. *The kernel k is a probability density function which is symmetric about 0, nondecreasing on $(-\infty,0]$ and nonincreasing on $[0,\infty)$, has a finite mode $k(0) < \infty$ and its second moment is strictly positive and finite. Thus, $\mu_0(k) = 1$, $\mu_1(k) = 0$, and $0 < \mu_2(k) < \infty$.*

Assumption 3.2.2. *The density f is at least four times differentiable and $R(f^{(r)}) < \infty$ for $r \leq 4$, where $f^{(r)}$ is the rth derivative of f.*

With MC, we have AIV $= n^{-1}h^{-1}\mu_0(k^2)$ and AISB $= (\mu_2(k))^2 R(f'')h^4/4$. The AMISE is minimized by taking $h^5 = Q/n$ where $Q := \mu_0(k^2)/[(\mu_2(k))^2 R(f'')]$, if Q is well-defined and finite. This gives

$$\text{AMISE} = (5/4)Q^{-1/5}\mu_0(k^2)n^{-4/5}.$$

Thus, finding a good h amounts to finding a good approximation of $R(f'')$. But since f is precisely the unknown function that we want to estimate, this seems to be a circular problem. However, perhaps surprisingly, a viable approach is to estimate $R(f'')$ by estimating f'' also via a KDE, integrating its square over $[a,b]$, and plugging this estimate into the formula for the optimal h (Berlinet and Devroye, 1994; Jones et al., 1996; Raykar and Duraiswami, 2006; Scott, 2015). To do that, one needs to select a good h to estimate f'' by a KDE. The asymptotically optimal h depends in turn on $R(f^{(4)})$ where $f^{(4)}$ is the fourth derivative of f. Then $R(f^{(4)})$ can be estimated by integrating the KDE of $f^{(4)}$ and this goes on ad infinitum. In practice, one can select an integer $r_0 \geq 1$, get a rough estimate of $R(f^{(r_0+2)})$, and start from there. One simple way of doing this is to pretend that f is a normal density with a mean and variance equal to the sample mean $\hat{\mu}$ and variance $\hat{\sigma}^2$ of the data, and then compute $R(f^{(r_0+2)})$ for this normal density. To estimate the rth derivative $f^{(r)}$, one can take the sample derivative of the KDE with a smooth kernel k, yielding

$$\hat{f}_n^{(r)}(x) \approx \frac{1}{nh^{r+1}} \sum_{i=0}^{n-1} k^{(r)}\left(\frac{x - X_i}{h}\right). \tag{3.2.1}$$

The asymptotically optimal h to use in this KDE is

$$h_*^{(r)} = \left(\frac{(2r+1)\mu_0((k^{(r)})^2)}{\mu_2^2(k)^2 R(f^{(r+2)})n}\right)^{1/(2r+5)}. \tag{3.2.2}$$

We will use this strategy to estimate a good h in our experiments with MC and RQMC, with a Gaussian kernel, with $r_0 = 2$.

In this paper we always take h to be the same for all $x \in [a,b]$. It is possible to improve the kernel density estimation by using a locally varying bandwidth $h(x) > 0$. For instance, it is advantageous to have a larger $h = h(x)$ where $f(x)$ is smaller. The interested reader is referred to Scott (2015); Terrell and Scott (1992).

3.3. Error and variance bounds for RQMC integration

We recall classical error and variance bounds for RQMC integration. They can be found in Dick and Pillichshammer (2010), L'Ecuyer (2009), Niederreiter (1992), and Owen (1997), for example. We will use them to obtain bounds on the AIV for the KDE.

The integration error of $g : [0,1]^s \to \mathbb{R}$ with the point set $P_n = \{\mathbf{u}_1, \ldots, \mathbf{u}_n\} \subset [0,1]^s$ is

$$E_n = \frac{1}{n}\sum_{i=1}^{n} g(\mathbf{u}_i) - \int_{[0,1]^s} g(\mathbf{u})\mathrm{d}\mathbf{u}.$$

Let $\mathfrak{v}$ denote a subset of coordinates, $\mathfrak{v} \subseteq \mathcal{S} := \{1, \ldots, s\}$. For any $\mathbf{u} = (u_1, \ldots, u_s) \in [0,1]^s$ we denote by $\mathbf{u}_\mathfrak{v}$ the projection of $\mathbf{u}$ to the coordinates in $\mathfrak{v}$ and by $(\mathbf{u}_\mathfrak{v}, \mathbf{1})$ the point $\mathbf{u}$ in which u_j has been replaced by 1 for each $j \notin \mathfrak{v}$. Furthermore, we write $g_\mathfrak{v} := \partial^{|\mathfrak{v}|} g / \partial \mathbf{u}_\mathfrak{v}$ for the partial derivative of g with respect to each of the coordinates in $\mathfrak{v}$. When $g_\mathfrak{v}$ exists and is continuous for $\mathfrak{v} = \mathcal{S}$, the *Hardy-Krause variation* of g is

$$V_{\mathrm{HK}}(g) = \sum_{\emptyset \neq \mathfrak{v} \subseteq \mathcal{S}} \int_{[0,1]^{|\mathfrak{v}|}} |g_\mathfrak{v}(\mathbf{u}_\mathfrak{v}, \mathbf{1})| \, \mathrm{d}\mathbf{u}_\mathfrak{v}. \tag{3.3.1}$$

The *star-discrepancy* of P_n is

$$D^*(P_n) = \sup_{\mathbf{u} \in [0,1]^s} \left| \mathrm{vol}[\mathbf{0}, \mathbf{u}) - \frac{|P_n \cap [\mathbf{0}, \mathbf{u})|}{n} \right|,$$

where $\mathrm{vol}[\mathbf{0}, \mathbf{u})$ is the volume of the box $[\mathbf{0}, \mathbf{u})$. The *Koksma-Hlawka inequality* states that

$$|E_n| \leq V_{\mathrm{HK}}(g) \cdot D^*(P_n). \tag{3.3.2}$$

Several known construction methods give P_n with $D^*(P_n) = \mathcal{O}((\log n)^{s-1}/n) = \mathcal{O}(n^{-1+\epsilon})$ for all $\epsilon > 0$. They include lattice rules and digital nets. Therefore, if $V_{\mathrm{HK}}(g) < \infty$, it is possible to achieve $|E_n| = \mathcal{O}(n^{-1+\epsilon})$ for the worst-case error. It is also known how to randomize the points of these constructions so that for the randomized points, $\mathbb{E}[E_n] = 0$ and

$$\mathrm{Var}[E_n] = \mathbb{E}[E_n^2] = \mathcal{O}(n^{-2+\epsilon}). \tag{3.3.3}$$

3.4. Bounding the convergence rate of the AIV for a KDE with RQMC

Replacing MC by RQMC does not affect the bias, because $\hat{f}_n(x)$ has the same expectation for both, but it can change the variance. Before trying to bound the variance under RQMC,

it is instructive to recall how it is bounded under MC. To compute (or bound) the IV over an interval $[a,b]$, we compute (or bound) $\mathrm{Var}[\hat{f}_n(x)]$ at an arbitrary point $x \in [a,b]$ and integrate this bound over x. Since $\hat{f}_n(x)$ is an average of n independent realizations of $Y(x) = k((x-X)/h)/h$, it suffices to compute $\mathrm{Var}[Y(x)]$ and we have $\mathrm{Var}[\hat{f}_n(x)] = \mathrm{Var}[Y(x)]/n$. With the change of variable $w = (x-v)/h$, we obtain (Scott, 2015, page 143):

$$\begin{aligned}\mathrm{Var}[Y(x)] &= \frac{1}{h^2}\int_{-\infty}^{\infty} k^2\left(\frac{x-v}{h}\right) f(v)\mathrm{d}v - \mathbb{E}^2[Y(x)] \\ &= \frac{1}{h}\int_{-\infty}^{\infty} k^2(w) f(x-hw)\mathrm{d}w - \mathbb{E}^2[Y(x)] = \frac{f(x)}{h}\int_{-\infty}^{\infty} k^2(w)\mathrm{d}w \; - f^2(x) + \mathcal{O}(h).\end{aligned}$$

Integrating over $x \in [a,b]$, gives $\mathrm{IV} = p_0\,\mu_0(k^2)/(nh) - R(f)/n + \mathcal{O}(h/n)$ where $p_0 = \int_a^b f(x)\mathrm{d}x \le 1$.

With RQMC, this also holds for a single RQMC point $\mathbf{U}_i$ and $X = X_i = g(\mathbf{U}_i)$, but to obtain $\mathrm{Var}[\hat{f}_n(x)]$, we can no longer just divide $\mathrm{Var}[Y(x)]$ by n, because the n realizations of $Y(x)$ are not independent. RQMC is effective if and only if these realizations are negatively correlated, in the sense that if $Y_i = h^{-1}k((x-X_i)/h)$, then $\sum_{i\neq j}\mathrm{Cov}(Y_i,Y_j) \le 0$. This would imply that RQMC can never be worse than MC, but this seems hard to prove.

We now take a different path, in which we examine how the KH inequality (4.3.3) can be used to bound $\mathrm{Var}[\hat{f}_n(x)]$. With $X = g(\mathbf{U})$, we can write

$$\hat{f}_n(x) = \frac{1}{n}\sum_{i=1}^{n}\tilde{g}(x,\mathbf{U}_i) \quad \text{where} \quad \tilde{g}(x,\mathbf{U}_i) := Y_i = \frac{1}{h}k\left(\frac{x-g(\mathbf{U}_i)}{h}\right). \tag{3.4.1}$$

Thus, $\hat{f}_n(x)$ can be interpreted as an RQMC estimator of $\mathbb{E}[\tilde{g}(x,\mathbf{U})] = \int_{[0,1]^s}\tilde{g}(x,\mathbf{u})\mathrm{d}\mathbf{u}$. To apply the bound in (4.3.4) to this estimator, we need to bound the variation of $\tilde{g}(x,\cdot)$, by bounding each term of the sum in (4.3.2).

To provide insight, we first examine the special case where $s = 1$ and g is nondecreasing over $[0,1]$, under Assumption 3.2.1. Then $\tilde{g}(x,u) = k((x-g(u))/h)/h$ is nonincreasing over the u with $g(u) \le x$ and nondecreasing over u with $g(u) \ge x$. In that case $V_{\mathrm{HK}}(\tilde{g}(x,\cdot))$ is the ordinary one-dimensional total variation, and then

$$V_{\mathrm{HK}}(\tilde{g}(x,\cdot)) \le \left|\frac{1}{h}k(0) - \frac{1}{h}k\left(\frac{x-g(0)}{h}\right)\right| + \left|\frac{1}{h}k(0) - \frac{1}{h}k\left(\frac{x-g(1)}{h}\right)\right| \le \frac{2k(0)}{h}. \tag{3.4.2}$$

The same bound holds for nonincreasing functions g. More generally, if g is monotone within each of M intervals that partition the domain $[0,1]$ then $V_{\mathrm{HK}}(\tilde{g}(x,\cdot)) \le 2Mk(0)/h$. The factor

of 2 is necessary because k is potentially increasing and then decreasing within each of those intervals. Note that there are one-dimensional point sets P_n with $D^*(P_n) \leq 1/n$. With such point sets, we obtain $\mathrm{Var}[\hat{f}_n(x)] \leq (2Mk(0))^2/(nh)^2$, so AIV $= \mathcal{O}((nh)^{-2})$. With $h = \Theta(n^{-1/3})$, this gives AMISE $= \mathcal{O}(n^{-4/3})$.

We now consider the general case $s \geq 1$. To bound $V_{\mathrm{HK}}(\tilde{g}(x,\cdot))$ we will make a similar change of variables as for the IV under MC. We need additional assumptions on k and g.

Assumption 3.4.1. *The kernel function $k : \mathbb{R} \to \mathbb{R}$ is s times continuously differentiable and its derivatives up to order s are integrable and uniformly bounded over $\mathbb{R}$.*

Assumption 3.4.2. *Let $g : [0,1]^s \to \mathbb{R}$ be piecewise monotone in each coordinate u_j when the other coordinates are fixed, with a number of monotone pieces (which is 1 plus the number of times that the function switches from strictly decreasing to strictly increasing or vice-versa, in u_j) that is bounded uniformly in $\mathbf{u}$ by an integer M_j. We also assume that the first-order partial derivatives of g are continuous and that $\|g_{\mathfrak{v}}\|_\infty < \infty$ for all $\mathfrak{v} \subseteq \mathcal{S}$. This implies that any product of partial derivatives of g of order at most one in each variable is integrable.*

Because the Hardy-Krause variation (4.3.2) involves mixed partial derivatives of $\tilde{g}(x,\cdot)$ of order up to s, things unfortunately become considerably more complicated than for MC. Roughly speaking, every derivative causes an additional factor h^{-1}, while we may dispose of only one such factor through a change of variables. This is reflected in Proposition 3.4.1 below. Similar to the one-dimensional case, we need to take into account how often g changes its monotonicity direction, and this is captured by the M_j's. For each $j \in \mathcal{S}$, let $G_j = \left\| \prod_{\ell \in \mathcal{S} \setminus \{j\}} g_{\{\ell\}} \right\|_\infty$ and $c_j = M_j \cdot \left(\|k^{(s)}\|_1 \cdot G_j + \mathbb{I}(s=2) \cdot \|k^{(s)}\|_\infty \cdot \|g_{\{1,2\}}\|_1 \right) < \infty$, where $\mathbb{I}(\cdot)$ is the indicator function, so the expression for c_j contains an extra term when $s = 2$. The source of this extra term is that for $s = 2$, the only partition of $\{1,2\}$ which contains no singletons is $\{1,2\}$ itself, and it gives a term in $h^{-2} = h^{-s}$, whereas for $s > 2$, all the extra terms are $\mathcal{O}(h^{-s+1})$. Our main result of this section is:

Proposition 3.4.1. *Let k and g satisfy Assumptions 3.2.1, 3.4.1 and 3.4.2, and $c = \min_{j \in \mathcal{S}} c_j$. . Then the Hardy-Krause variation of the function $\tilde{g}(x,\mathbf{u}) = h^{-1}k((x - g(\mathbf{u}))h^{-1})$ (as a function of $\mathbf{u}$) satisfies*

$$V_{\mathrm{HK}}(\tilde{g}(x,\cdot)) \leq ch^{-s} + \mathcal{O}(h^{-s+1}). \tag{3.4.3}$$

Note that the constants c_j depend on g via M_j and G_j (plus an extra term when $s = 2$). A large M_j means that g changes the sign of its slope many times in the direction of coordinate j. A large G_j means that the product of the slopes in the directions of the other coordinates can attain large values. When we have both, then c_j is large. Observe that the bound (3.4.3) uses the *smallest* c_j. In case g is constant with respect to one coordinate $\ell \neq j$, then $G_j = 0$, $c_j = 0$, and $c = 0$. That is, the term in h^{-s} disappears from (3.4.3) and the bound becomes $\mathcal{O}(h^{-s+1})$. This agrees with the fact that g is then effectively a $(s-1)$-dimensional function. Likewise, if g is almost constant (has very little variation) with respect to one or more coordinate(s) $\ell \neq j$, then G_j and therefore c_j will be small, unless the other terms in the product are very large. Before proving this proposition, we state a corollary that bounds the AIV and the AMISE rates under RQMC.

Corollary 3.4.1. *Let k and g satisfy Assumptions 3.2.1, 3.2.2, 3.4.1 and 3.4.2. For a KDE with kernel k, with the underlying observations obtained via sets P_n of n RQMC points for which $D^*(P_n) = \mathcal{O}(n^{-1+\epsilon})$ for all $\epsilon > 0$ when $n \to \infty$, by combining (3.4.3) with (4.3.3) and squaring, we find that* $\text{AIV} = \mathcal{O}(n^{-2+\epsilon}h^{-2s})$ *for all $\epsilon > 0$. Then, by taking $h = \Theta(n^{-1/(2+s)})$, we obtain that* $\text{AMISE} = \mathcal{O}(n^{-4/(2+s)+\epsilon})$ *for all $\epsilon > 0$. The exponent of n in this AMISE bound beats the MC rate for $s < 3$ and is almost equal to the MC rate for $s = 3$.*

Let $\Pi(\mathfrak{v})$ denote the set of all partitions of a set of coordinate indices $\mathfrak{v} \subseteq \mathcal{S}$, and let $\Pi_1(\mathfrak{v})$ denote the subset of all partitions that contain at least one singleton. For each partition $P \in \Pi_1(\mathfrak{v})$, we select a particular singleton and denote it by $\{j(P)\}$. Removing that singleton from P yields a partition of $\mathfrak{v}^* = \mathfrak{v} \setminus \{j(P)\}$ which we denote by P^*.

The proof of the proposition will use the following lemma, which describes when the aforementioned change of variable works and how it removes a factor $1/h$ from the bound.

Lemma 3.4.1. *Let Assumptions 3.2.1, 3.4.1 and 3.4.2 hold, let $h > 0$, $\mathfrak{v} \subseteq \mathcal{S}$, and $P \in \Pi_1(\mathfrak{v})$. Then*

$$\int_{[0,1]^{|\mathfrak{v}|}} \left| k^{(|P|)}\left(\frac{x - g(\mathbf{u}_\mathfrak{v}, \mathbf{1})}{h}\right) \cdot \prod_{\mathfrak{w} \in P} g_\mathfrak{w}(\mathbf{u}_\mathfrak{v}, \mathbf{1}) \right| \mathrm{d}\mathbf{u}_\mathfrak{v} \quad \leq \quad h \cdot M_{j(P)} \cdot \left\| \prod_{\mathfrak{w} \in P^*} g_\mathfrak{w} \right\|_\infty \cdot \|k^{(|P|)}\|_1.$$

Proof. We assume without loss of generality that $1 \in \mathfrak{v}$ and $j(P) = 1$. We make the change of variables

$$u_1 \mapsto w = (x - g(\mathbf{u}_\mathfrak{v}, \mathbf{1}))/h. \tag{3.4.4}$$

For any $\mathbf{u}_\mathfrak{v} \in [0,1]^{|\mathfrak{v}|}$ with fixed $\mathbf{u}_{\mathfrak{v}^*} \in [0,1]^{|\mathfrak{v}|-1}$, we partition $[0,1]$ into a part $\mathcal{N}(\mathbf{u}_{\mathfrak{v}^*})$ where $g(\mathbf{u}_\mathfrak{v},\mathbf{1})$ is constant in u_1 and into sets $\mathcal{D}_l(\mathbf{u}_{\mathfrak{v}^*})$, $1 \leq l \leq L(\mathbf{u}_{\mathfrak{v}^*}) \leq M_1$, on which it is either strictly decreasing or strictly increasing in u_1. Since $g_{\{1\}}$ is continuous by assumption, each of these sets is measurable. The restriction of $g_{\{1\}}$ to $\mathcal{N}(\mathbf{u}_{\mathfrak{v}^*})$ equals 0 identically.

In all the other sets $\mathcal{D}_l(\mathbf{u}_{\mathfrak{v}^*})$ we apply the change of variables (3.4.4). Considering this in the left-hand side of the claim leads to

$$\begin{aligned}
&\int_{[0,1]^{|\mathfrak{v}|}} \left| k^{(|P|)} \left(\frac{x - g(\mathbf{u}_\mathfrak{v},\mathbf{1})}{h} \right) \prod_{\mathfrak{w} \in P} g_\mathfrak{w}(\mathbf{u}_\mathfrak{v},\mathbf{1}) \right| \mathrm{d}\mathbf{u}_\mathfrak{v} \\
&\quad = \int_{[0,1]^{|\mathfrak{v}|-1}} \sum_{l=1}^{L(\mathbf{u}_{\mathfrak{v}^*})} \int_{\mathcal{D}_l(\mathbf{u}_{\mathfrak{v}^*})} \left| k^{(|P|)} \left(\frac{x - g(\mathbf{u}_\mathfrak{v},\mathbf{1})}{h} \right) \prod_{\mathfrak{w} \in P} g_\mathfrak{w}(\mathbf{u}_\mathfrak{v},\mathbf{1}) \right| \mathrm{d}u_1 \mathrm{d}\mathbf{u}_{\mathfrak{v}^*} \\
&\quad \leq h \int_{[0,1]^{|\mathfrak{v}|-1}} L(\mathbf{u}_{\mathfrak{v}^*}) \int_{-\infty}^{\infty} \left| k^{(|P|)}(w) \prod_{\mathfrak{w} \in P^*} g_\mathfrak{w}(\mathbf{u}_\mathfrak{v},\mathbf{1}) \right| \mathrm{d}w \mathrm{d}\mathbf{u}_{\mathfrak{v}^*} \\
&\quad \leq h \cdot M_1 \cdot \|k^{(|P|)}\|_1 \cdot \left\| \prod_{\mathfrak{w} \in P^*} g_\mathfrak{w} \right\|_\infty ,
\end{aligned}$$

where we used Hölder's inequality in the last step. □

Proof of Proposition 3.4.1. We rewrite each summand w.r.t. $\mathfrak{v} \subseteq \mathcal{S}$ in (4.3.2) with the help of Faà di Bruno's formula (see (Hardy, 2006, Proposition 1)) as follows

$$\begin{aligned}
\int_{[0,1]^{|\mathfrak{v}|}} |\tilde{g}_\mathfrak{v}(x,\mathbf{u}_\mathfrak{v},\mathbf{1})| \, \mathrm{d}\mathbf{u}_\mathfrak{v} &= \frac{1}{h} \int_{[0,1]^{|\mathfrak{v}|}} \left| \sum_{P \in \Pi(\mathfrak{v})} k^{(|P|)} \left(\frac{x - g(\mathbf{u}_\mathfrak{v},\mathbf{1})}{h} \right) \prod_{\mathfrak{w} \in P} \frac{\partial^{|\mathfrak{w}|}}{\partial \mathbf{u}_\mathfrak{w}} \left(\frac{x - g(\mathbf{u}_\mathfrak{v},\mathbf{1})}{h} \right) \right| \mathrm{d}\mathbf{u}_\mathfrak{v} \\
&\leq \sum_{P \in \Pi(\mathfrak{v})} \frac{1}{h^{|P|+1}} \int_{[0,1]^{|\mathfrak{v}|}} \left| k^{(|P|)} \left(\frac{x - g(\mathbf{u}_\mathfrak{v},\mathbf{1})}{h} \right) \prod_{\mathfrak{w} \in P} g_\mathfrak{w}(\mathbf{u}_\mathfrak{v},\mathbf{1}) \right| \mathrm{d}\mathbf{u}_\mathfrak{v}. \qquad (3.4.5)
\end{aligned}$$

If $P \in \Pi_1(\mathfrak{v})$, we bound the corresponding summand in (3.4.5) via Lemma 3.4.1. If $P \notin \Pi_1(\mathfrak{v})$, we apply Hölder's inequality to obtain the upper bound

$$\frac{1}{h^{|P|+1}} \cdot \|k^{(|P|)}\|_\infty \cdot \left\| \prod_{\mathfrak{w} \in P} g_\mathfrak{w} \right\|_1 .$$

Furthermore, we observe that each element of $P \in \Pi(\mathfrak{v}) \setminus \Pi_1(\mathfrak{v})$ has a cardinality of at least 2. Therefore, P can contain at most $\lfloor |\mathfrak{v}|/2 \rfloor$ elements. This gives the following bound on

$V_{\mathrm{HK}}(\tilde{g}(x,\cdot))$, which holds for any $j \in \mathcal{S}$:

$$\begin{aligned} V_{\mathrm{HK}}(\tilde{g}(x,\cdot)) &\le \sum_{\emptyset\neq\mathfrak{v}\subseteq\mathcal{S}} \left[\sum_{P\in\Pi_1(\mathfrak{v})} \frac{M_{j(P)}}{h^{|P|}} \|k^{(|P|)}\|_1 \cdot \Big\| \prod_{\mathfrak{w}\in P^*} g_{\mathfrak{w}} \Big\|_\infty + \sum_{P\in\Pi(\mathfrak{v})\setminus\Pi_1(\mathfrak{v})} \frac{1}{h^{|P|+1}} \|k^{(|P|)}\|_\infty \cdot \Big\| \prod_{\mathfrak{w}\in P} g_{\mathfrak{w}} \Big\|_1 \right] \\ &\le h^{-s} M_j G_j \, \|k^{(s)}\|_\infty + \mathcal{O}(h^{-s+1}) + \sum_{\emptyset\neq\mathfrak{v}\subseteq\mathcal{S}} h^{-\lfloor|\mathfrak{v}|/2\rfloor-1} \sum_{P\in\Pi(\mathfrak{v})\setminus\Pi_1(\mathfrak{v})} \|k^{(|P|)}\|_\infty \cdot \Big\| \prod_{\mathfrak{w}\in P} g_{\mathfrak{w}} \Big\|_1. \end{aligned}$$

For $s = 1$ this already proves the claim.

For $s = 2$, the only partition of $\mathcal{S}$ that contains no singleton is $\mathcal{S}$ itself and the result follows. For $s \ge 3$ we have $\lfloor s/2 \rfloor + 1 \le s - 1$, and then

$$\sum_{\emptyset\neq\mathfrak{v}\subseteq\mathcal{S}} h^{-\lfloor|\mathfrak{v}|/2\rfloor-1} \sum_{P\in\Pi(\mathfrak{v})\setminus\Pi_1(\mathfrak{v})} \|k^{(|P|)}\|_\infty \cdot \Big\| \prod_{\mathfrak{w}\in P} g_{\mathfrak{w}} \Big\|_1 = \mathcal{O}(h^{-s+1}).$$

□

The bound of Proposition 3.4.1 suggests that the IV could converge at a much worse rate with RQMC than with MC when s is large. However, the next proposition, based on a result of Owen (1998c), provides a different bound that does not grow as h^{-2s} when h decreases, for a particular type of RQMC point set, namely a (t,m,s)-net in base 2 randomized by a nested uniform scramble (NUS). This type of point set contains 2^m points in s dimensions, the t parameter measures the uniformity in some sence (the smaller the better) (Dick and Pillichshammer, 2010; Niederreiter, 1992), and the NUS shuffles the points in some particular way (Owen, 1995, 1997). We state the following result for base $b = 2$, but it can be extended to a general prime base $b \ge 2$.

Proposition 3.4.2. *Let P_n be a (t,m,s)-net in base 2 randomized by NUS, and let Assumption 3.2.1 hold. Then the IV of $\hat{f}_n$ satisfies*

$$\mathrm{IV} \le 2^t 3^s \mu_0(k^2)/(nh) + \mathcal{O}(h/n).$$

Moreover for any fixed $s \ge 1$, there is a fixed $t \ge 0$ for which we know how to construct a (t,m,s)-net P_n in base 2 for any integer $m \ge 1$. By using such a sequence of nets with NUS, we get $\mathrm{IV} = \mathcal{O}(1/(nh))$, and then by taking $h = \Theta(n^{-1/5})$, we obtain $\mathrm{MISE} = \mathcal{O}(n^{-4/5})$. That is, the MISE never converges at a worse rate than with plain MC.

Proof. Let $\mathrm{Var}_{\mathrm{MC}}$ and $\mathrm{Var}_{\mathrm{NUS}}$ denote the variance under MC and under NUS, respectively. Likewise, for any given pair (n,h), let $\mathrm{IV}_{\mathrm{MC}}(n,h)$ and $\mathrm{IV}_{\mathrm{NUS}}(n,h)$ denote the IV under MC

and under NUS, respectively, and similarly for the MISE. Under Assumption 3.2.1, $\tilde{g}(x,\cdot)$ is square-integrable over $[0,1]^s$ for any $x \in [a,b]$, so we can apply Theorem 1 of Owen (1998c), which tells us that

$$\mathrm{Var}_{\mathrm{NUS}}[\tilde{g}(x,\mathbf{U})] \;\le\; 2^t 3^s \mathrm{Var}_{\mathrm{MC}}[\tilde{g}(x,\mathbf{U})].$$

By integrating, we obtain $\mathrm{IV}_{\mathrm{NUS}}(n,h) \le 2^t 3^s \mathrm{IV}_{\mathrm{MC}}(n,h)$. We saw earlier that $\mathrm{IV}_{\mathrm{MC}}(n,h) \le \mu_0(k^2)/(nh) - R(f)/n + \mathcal{O}(h/n)$, and this proves the displayed inequality.

For the second part, for any $s \ge 1$, there is a fixed $t \ge 0$ for which we know how to construct a (t,s)-sequence in base 2; see Sobol' (1967) and (Niederreiter, 1992, Section 4.5), for example. For any integer m, the first 2^m points of such a sequence form a (t,m,s)-net in base 2. Thus, t can be assumed to be bounded uniformly in m, and therefore $\mathrm{IV}_{\mathrm{NUS}}(n,h) = \mathcal{O}(\mathrm{IV}_{\mathrm{MC}}(n,h)) = \mathcal{O}(1/(nh))$. Since MC and NUS give the same ISB, we also have $\mathrm{MISE}_{\mathrm{NUS}}(n,h) \le 2^t 3^s \mathrm{MISE}_{\mathrm{MC}}(n,h) = \mathcal{O}(1/(nh) + h^4) = \mathcal{O}(h^{-4/5})$ if we take $h = \Theta(n^{-1/5})$.

□

3.5. Stratified sampling of $[0,1)^s$

In this section, we examine how plain stratified sampling of the unit hypercube can reduce the IV of the KDE compared with MC. We consider point sets P_n constructed as in Assumption 3.5.1 below. This type of stratified sampling can never increase the IV compared with MC. We prove this via a standard variance decomposition argument. Then, under the additional condition that $g(\mathbf{u})$ is monotone with respect to each coordinate of $\mathbf{u}$, we prove an IV bound that converges at a faster rate than the IV under MC when $s < 5$. The KH inequality and the variation of g are not involved in the IV bound developed here; we work directly with the variance. For this reason, the bound will not contain the annoying factor h^{-2s} as in Proposition 3.4.1. On the other hand, the exponent of n will not be as good. Our main results are Propositions 3.5.1 and 3.5.2, and Corollary 3.5.1.

Assumption 3.5.1. *The hypercube $[0,1)^s$ is partitioned into $n = q^s$ congruent cubic cells $S_{\mathbf{i}} := \prod_{j=1}^s [i_j/q,(i_j+1)/q)$, $\mathbf{i} \in \mathbf{I} = \{\mathbf{i} = (i_1,i_2,\ldots,i_s) : 0 \le i_j < q$ for each $j\}$, for some integer $q \ge 2$. We construct $P_n = \{\mathbf{U}_1,\ldots,\mathbf{U}_n\}$ by sampling one point uniformly in each subcube $S_{\mathbf{i}}$, independently across the subcubes, and put $X_i = g(\mathbf{U}_i)$ for $i = 1,\ldots,n$.*

Proposition 3.5.1. *Under Assumptions 3.2.1 and 3.5.1, the IV of a KDE $\hat{f}_n$ with kernel k never exceeds the IV of the same estimator under standard MC, which satisfies* $\text{IV} \leq \mu_0(k^2)/(nh) - R(f)/n + \mathcal{O}(h/n)$.

Proof. We can decompose the variance under MC as

$$\begin{aligned}\text{Var}[\tilde{g}(x,\mathbf{U})] &= \mathbb{E}[\text{Var}[\tilde{g}(x,\mathbf{U}) \mid \mathbf{U} \in S_{\mathbf{i}}] + \text{Var}[\mathbb{E}[\tilde{g}(x,\mathbf{U}) \mid \mathbf{U} \in S_{\mathbf{i}}] \\ &= \frac{1}{n}\sum_{\mathbf{i}\in\mathbf{I}} \text{Var}[\tilde{g}(x,\mathbf{U}) \mid \mathbf{U} \in S_{\mathbf{i}}] + \frac{1}{n}\sum_{\mathbf{i}\in\mathbf{I}} (\mu_{\mathbf{i}} - \mu)^2,\end{aligned}$$

where $\mu = \mathbb{E}[\tilde{g}(x,\mathbf{U})]$ and $\mu_{\mathbf{i}} = \mathbb{E}[\tilde{g}(x,\mathbf{U}) \mid \mathbf{U} \in S_{\mathbf{i}}]$. By sampling exactly one point in each cell $S_{\mathbf{i}}$, the stratified sampling removes the second term, and the first term remains the same. Therefore, stratification never increases $\text{Var}[\hat{f}_n(x)]$. The second term also indicates how the amount of variance reduction depends on how the $\mu_{\mathbf{i}}$ vary between boxes. □

Now we know that stratification can do no harm. To show that it can also improve the convergence rate of the MISE, we need additional conditions.

Assumption 3.5.2. *For each $j \in \mathcal{S}$, the function $g : [0,1)^s \to \mathbb{R}$ is* monotone *in u_j when the other $s-1$ coordinates are fixed, and the direction of monotonicity in u_j (nondecreasing or nonincreasing) is the same for all values of the other coordinates. Without loss of generality, we will assume in the rest of the paper that it is nondecreasing in each coordinate. (If it is nonincreasing in u_j, one can simply replace u_j by $1-u_j$ in the definition of g and this does not change the distribution of $X = g(\mathbf{U})$.)*

Proposition 3.5.2. *Let Assumptions 3.2.1, 3.5.1 and 3.5.2 hold and let $\hat{f}_n$ be a KDE with kernel k obtained from $X_1, X_2, \ldots, X_n$. Then the IV of $\hat{f}_n$ satisfies*

$$\text{IV} \leq (b-a)s \cdot k^2(0) \cdot h^{-2} n^{-(s+1)/s}.$$

Corollary 3.5.1. *Under Assumptions 3.2.1, 3.2.2, 3.5.1 and 3.5.2, the AMISE bound is minimized by taking $h = \kappa n^{-(s+1)/(6s)}$ with $\kappa^6 = [(b-a)s \cdot k^2(0)]/[(\mu_2(k))^2 R(f'')/2]$, and this gives* $\text{AMISE} = Kn^{-\nu}$ *with $\nu = (2/3)(s+1)/s$ and $K = (b-a)s \cdot k^2(0)\kappa^{-2} + (\mu_2(k))^2 R(f'')\kappa^4/4$. The exponent of n in this bound beats the MC rate for all $s < 5$ and is equal to the MC rate for $s = 5$.*

The corollary is straightforward to prove. It suffices to minimize with respect to h the sum of the AISB (given in Section 3.2) and the IV bound.

Our proof of Proposition 3.5.2 is inspired by (L'Ecuyer et al., 2008, Proposition 6). It combines three lemmas that we state and prove before proving the proposition. These lemmas could be useful in other contexts as well. The first lemma bounds the variance of the KDE in terms of a uniform bound K_n on the variance of the empirical cdf estimator. The other lemmas bound K_n.

Let F be the cdf of the random variable X. For any $x \in \mathbb{R}$, the empirical cdf of a sample $X_1, X_2 \ldots, X_n$ evaluated at x is $\hat{F}_n(x) := n^{-1} \sum_{i=1}^{n} \mathbb{I}[X_i \le x]$. We denote the difference by $\Delta_n(x) = \hat{F}_n(x) - F(x)$. Let $K_n := \sup_{x \in \mathbb{R}} \mathrm{Var}[\Delta_n(x)] = \sup_{x \in \mathbb{R}} \mathrm{Var}[\hat{F}_n(x)]$.

Lemma 3.5.1. *Under Assumption 3.2.1, for all $x \in \mathbb{R}$, we have*

$$\mathrm{Var}[\hat{f}_n(x)] \le 2h^{-2} k(0) K_n.$$

Proof. We will prove the inequality

$$\mathrm{Var}[\hat{f}_n(x)] \le K_n h^{-4} \left(\int_{\mathbb{R}} k'\left(\frac{x-z}{h}\right) \mathrm{d}z \right)^2. \tag{3.5.1}$$

The result follows from this inequality by making the change of variable $w = (x - z)/h$ in the integral. To prove (3.5.1), we rewrite the variance of $\hat{f}_n(x)$ using integration by parts as follows:

$$\begin{aligned}
\mathbb{E}\left[\left|\hat{f}_n(x) - \mathbb{E}[\hat{f}_n(x)]\right|^2\right] &= h^{-2}\, \mathbb{E}\left[\left| \int_{\mathbb{R}} k\left(\frac{x-z}{h}\right) \mathrm{d}\hat{F}_n(z) - \int_{\mathbb{R}} k\left(\frac{x-z}{h}\right) \mathrm{d}F(z) \right|^2\right] \\
&= h^{-4}\, \mathbb{E}\left[\left| \int_{\mathbb{R}} \left(\hat{F}_n(z) - F(z)\right) k'\left(\frac{x-z}{h}\right) \mathrm{d}z \right|^2\right] \\
&= h^{-4}\, \mathbb{E}\left[\int_{\mathbb{R}} \int_{\mathbb{R}} \Delta_n(y)\Delta_n(z) k'\left(\frac{x-y}{h}\right) k'\left(\frac{x-z}{h}\right) \mathrm{d}y \mathrm{d}z\right].
\end{aligned} \tag{3.5.2}$$

Observe that $\mathbb{E}[\Delta_n(z)] = 0$ since $\mathbb{E}[\hat{F}_n(z)] = F(z)$. Consequently,

$$\mathbb{E}[\Delta_n(y)\Delta_n(z)] = \mathrm{Cov}[\Delta_n(y), \Delta_n(z)] \le K_n.$$

Finally, (3.5.1) follows by interchanging the expectation and the integrals in (3.5.2). □

For all $x \in \mathbb{R}$, define $\mathbf{H}(x) = \{\mathbf{u} \in [0,1)^s : g(\mathbf{u}) \le x\}$ and its complement $\overline{\mathbf{H}}(x) = [0,1)^s \setminus \mathbf{H}(x)$. Under Assumption 3.5.1, let $\mathcal{B}(x)$ be the set of subcubes $S_{\mathbf{i}}$ that have a nonempty intersection with both $\mathbf{H}(x)$ and $\overline{\mathbf{H}}(x)$. The next lemma bounds the cardinality of $\mathcal{B}(x)$ when g is nondecreasing.

Lemma 3.5.2. *Under Assumptions 3.5.1 and 3.5.2, for all $x \in \mathbb{R}$, $|\mathcal{B}(x)| \leq sn^{(s-1)/s}$.*

PROOF. Let $\mathbf{I}_0 = \{\mathbf{i} = (i_1,i_2,\ldots,i_s) \in \mathbf{I}$ such that $\min_j i_j = 0\}$, which is the set of indices $\mathbf{i} \in \mathbf{I}$ for which at least one face of $S_\mathbf{i}$ lies on a face of $[0,1)^s$ that contains the origin. Each of the s faces of $[0,1)^s$ touches at most b^{s-1} elements of $\mathbf{I}_0$ and therefore $|\mathbf{I}_0| \leq sq^{s-1} = sn^{(s-1)/s}$. Now, for any $\mathbf{i} = (i_1,i_2,\ldots,i_s) \in \mathbf{I}_0$, consider the *diagonal string* of subcubes $S_{\mathbf{i}'(k)}$ with $\mathbf{i}'(k) = (i_1+k,i_2+k,\ldots,i_s+k)$ for $0 \leq k < q-\max_j i_j$. We argue that for any $x \in \mathbb{R}$, at most one subcube in this diagonal string can belong to $\mathcal{B}(x)$. Indeed, suppose that two distinct subcubes in the string belong to $\mathcal{B}(x)$, say $S_{\mathbf{i}'(k_1)}$ and $S_{\mathbf{i}'(k_2)}$ for $k_1 < k_2$. Since both subcubes contain points from $\mathbf{H}(x)$ and $\overline{\mathbf{H}}(x)$, there must be two points $\mathbf{u}_1 \in S_{\mathbf{i}'(k_1)} \cap \overline{\mathbf{H}}(x)$ and $\mathbf{u}_2 \in S_{\mathbf{i}'(k_2)} \cap \mathbf{H}(x)$. This implies that $g(\mathbf{u}_2) \leq x < g(\mathbf{u}_1)$ while $\mathbf{u}_1 < \mathbf{u}_2$ coordinatewise, which contradicts the assumption that g is nondecreasing. Since there are no more than $sn^{(s-1)/s}$ diagonal strings and each contains at most one element of $\mathcal{B}(x)$, the result follows. □

Lemma 3.5.3. *Under Assumption 3.5.1 and 3.5.2, $K_n \leq (s/4)h^{-2}n^{-(s+1)/s}$.*

PROOF. For each $\mathbf{i} \in \mathbf{I}$, consider the random variables

$$\delta_\mathbf{i}(x) = |P_n \cap \mathbf{H}(x) \cap S_\mathbf{i}| - n\,\mathrm{vol}(\mathbf{H}(x) \cap S_\mathbf{i}).$$

We make three observations. Firstly, $S_\mathbf{i}$ contains exactly one point of P_n by Assumption 3.5.1. Consequently, each $\delta_\mathbf{i}(x)$ is a Bernoulli random variable (with parameter $p = n\,\mathrm{vol}(\mathbf{H}(x) \cap S_\mathbf{i})$) minus its mean p and, therefore, $\mathrm{Var}[\delta_\mathbf{i}(x)] = p(1-p) \leq 1/4$. Secondly, for each $\mathbf{i}$ for which $S_\mathbf{i} \notin \mathcal{B}(x)$, $\delta_\mathbf{i}(x) = 0$, so $\mathrm{Var}[\delta_\mathbf{i}(x)] = 0$. Thirdly, for any two distinct subcubes, the positions of the points of P_n in these subcubes are independent. As a consequence of these three observations we see that

$$\mathrm{Var}[\Delta_n(x)] = \mathrm{Var}\left[\frac{1}{n}\sum_{\mathbf{i}\in\mathbf{I}} \delta_\mathbf{i}(x)\right] = \frac{1}{n^2}\sum_{\mathbf{i}:S_\mathbf{i}\in\mathcal{B}(x)} \mathrm{Var}[\delta_\mathbf{i}(x)] \leq \frac{1}{4n^2} sn^{-(s+1)/s}.$$

By applying Lemmas 3.5.1 and 3.5.2 we then obtain

$$\mathrm{Var}[\hat{f}_n(x)] \;\leq\; 2h^{-2}k(0)K_n \;\leq\; \frac{k(0)}{2(hn)^2}|\mathcal{B}(x)| \;\leq\; \frac{sk(0)}{2}h^{-2}n^{-(s+1)/s}.$$

□

Proof of Proposition 3.5.2. Combining Lemmas 3.5.1 and 3.5.3 and integrating the variance bound with respect to x over $[a,b]$ yields the result. □

In the above arguments, we assumed that the strata were cubic, but this is not necessary. We could instead partition $[0,1)^s$ into $n = \prod_{j=1}^{s} q_j$ cells congruent to $\prod_{j=1}^{d}[0,1/q_j)$, subject to the condition $\max_j q_j \le \lambda \min_j q_j$ for some $\lambda < \infty$. Then one can bound the cardinality of $\mathcal{B}(x)$ and $\mathrm{Var}[\hat{F}_n(x)]$ in a similar way. Non-cubic strata make sense if g varies more in some directions than in others. Finally, our bounds are proved under the assumption that g is monotone, but this assumption is *not necessary* for stratification to improve the MISE and/or its convergence rate.

3.6. Empirical Study

Our analysis in the previous sections was in terms of (asymptotic) bounds. Here, we study the IV and MISE behavior from a different viewpoint: our goal is to estimate empirically how they really behave in a range of values of n and h that one is likely to use. For this, we use a simple regression model to approximate the true IV and MISE in the region of interest. For some examples, we estimate the model parameters from simulated data, test the goodness of fit of the regression models in-sample and out-of-sample, and show how the model permits one to estimate the optimal h as a function of n, as well as the resulting MISE and its convergence rate, under RQMC.

3.6.1. Experimental setting and regression models for the local behavior of the IV, ISB, and MISE

We will use the following models to approximate the true IV and ISB in a limited range of values of n and h of interest:

$$\mathrm{IV} \approx Cn^{-\beta}h^{-\delta} \quad \text{and} \quad \mathrm{ISB} \approx Bh^{\alpha}, \tag{3.6.1}$$

for positive constants C, β, δ, and B, that can be estimated as explained below, and with $\alpha = 4$. This gives $\mathrm{MISE} \approx Cn^{-\beta}h^{-\delta} + Bh^{\alpha}$. The bounds derived in the previous sections have this form, and this motivates our model, but here we want to estimate the true values, which generally differ from the bounds. Once the parameters are estimated, we can estimate the optimal h by minimizing the MISE estimate for any given n in the selected range. In our

setting, this estimated MISE is a convex function of h. Taking the derivative with respect to h and setting it to zero yields $h^{\alpha+\delta} = [C\delta/(B\alpha)]n^{-\beta}$. Thus, if we take $h = \kappa n^{-\gamma}$, the constants κ and γ that minimize the MISE (based on our model) are $\kappa = \kappa_* := (C\delta/B\alpha)^{1/(\alpha+\delta)}$ and $\gamma = \gamma_* := \beta/(\alpha+\delta)$. Plugging them into the MISE expression gives

$$\text{MISE} \approx Kn^{-\nu} \tag{3.6.2}$$

with $K = K_* := C\kappa_*^{-\delta} + B\kappa_*^{\alpha}$ and $\nu = \nu_* := \alpha\beta/(\alpha+\delta)$. If h is taken too small (e.g., by taking $\kappa < \kappa_*$ or $\gamma > \gamma_*$ in the formula for h), the IV will be too large and will dominate the MISE, so we will observe a MISE that decreases just like the IV. The opposite happens if h is too large: the ISB dominates the MISE.

To estimate the model parameters for IV, we take the log to obtain the linear model

$$\log(\text{IV}) \approx \log C - \beta \log n - \delta \log h, \tag{3.6.3}$$

and we estimate the parameters C, β, and δ by linear regression. Since n is always a power of 2 for our RQMC points, we take all the logarithms in base 2. In our experiments, we selected a set of 36 pairs (n,h) with $n = 2^{14}, \ldots, 2^{19}$ and $h = h_0, \ldots, h_5$ where $h_j = h_0 2^{j/2} = 2^{-\ell_0 + j/2}$ and $2\ell_0$ is an integer selected from pilot runs (see the supplement for the details). This selection of ℓ_0 is the only step that requires human intervention.

For each n and each point set (MC, Stratification or RQMC), we generate a sample of size n, sort the sample, and then compute the density estimator for each h, for this sample. That is, we use the same sample for all estimation methods and all h. We make $n_r = 100$ independent replications of this procedure, which gives us independent replicates of the density estimator for the selected pairs (n,h). To obtain an unbiased estimator of the integral that defines the IV, we take a stratified sample of $n_e = 1024$ evaluation points over the interval $[a,b]$, compute the empirical variance of the KDE at each point, based on the n_r replications, and take the average multiplied by $(b-a)$. Larger values of n_e gave about the same estimates.

We approximate the ISB in (3.6.1) by the AISB, for which $\alpha = 4$ and $B = (\mu_2(k))^2 R(f'')/4$. For the Gaussian kernel, used in all our experiments, Assumptions 3.2.1 and 3.4.1 are satisfied, and $\mu_2(k) = 1$. We estimate the integral $R(f'')$ as explained in Section 3.2, using RQMC instead of MC to improve the accuracy.

Once we have the estimates $\hat{\kappa}_*$ and $\hat{\gamma}_*$ of κ_* and γ_*, we test the models out-of-sample by making an independent set of simulation experiments with pairs (n,h) that satisfy $h = \hat{h}_*(n) := \hat{\kappa}_* n^{-\hat{\gamma}_*}$ (the estimated optimal h) for a series of values of n. At each of these pairs (n,h), we sample n_r fresh independent replicates of the RQMC density estimator and compute the IV estimate, as well as the MISE estimate in the simple examples where the density is known. In the latter case, we fit again the linear regression model for log(MISE) vs $\log n$ to re-estimate the parameters K and ν in (3.6.2) and assess the goodness-of-fit. In our results, we denote these new estimates by $\tilde{K}$ and $\tilde{\nu}$. Of course, these model testing steps are not needed if one wishes to only estimate the density f and not to study the convergence properties.

In the end, we also compare the efficiencies of different methods for the same example by comparing their estimated MISE for $n = 2^{19}$ with the h recommended by the model. We denote by e19 the value of $-\log_2(\text{MISE})$ for $n = 2^{19}$; that is, we have MISE $= 2^{-e19}$. The efficiency gain of RQMC vs MC can be assessed by comparing their e19 values.

The point sets considered in our experiments were: (1) independent points (MC); (2) stratification of the unit cube (Stratif); (3) a Sobol' point set with a left random matrix scrambling and random digital shift (Sobol'+LMS); and (4) a Sobol' point set with nested uniform scrambling (Sobol'+NUS). The last two are well-known RQMC point sets (L'Ecuyer, 2018; Owen, 1997, 2003) and we view stratification as a weak form of RQMC. The short names in parentheses are used in the plots and tables. These point sets and randomizations are implemented in SSJ (L'Ecuyer, 2016), which we used for our experiments. More details and results can be found in the supplement.

3.6.2. A normalized sum of standard normals

As in Owen (2017), we construct a set of test functions with arbitrary dimension s and for which the density f of X is always the standard normal, $f(x) = \exp(-x^2/2)/\sqrt{2\pi}$ for any s. For this, let $Z_1, \ldots, Z_s$ be s independent standard normal random variables generated by inversion and put $X = (a_1 Z_1 + \cdots + a_s Z_s)/\sigma$, where $\sigma^2 = a_1^2 + \cdots + a_s^2$. For this simple example, the density is already known, so there is no need to estimate it, but this is convenient for testing the methodology, since it permits us to compute and compare unbiased estimators of the IV, ISB, and MISE for both MC and RQMC. For MC, these quantities do

not depend on s, but for RQMC, the IV and MISE do depend on s, and we want to see in what way.

We can also compute $R(f'')$ exactly in this example, which means we can compute B for the AISB and the asymptotically optimal h for the AMISE. However, we will first make experiments as if we did not know this B and have to estimate it, and then compare our estimates with the exact B. Here, g is a monotone increasing function, so Corollary 3.5.1 applies when we use stratification. Assumption 3.4.2 holds only if we truncate the normal distributions of the Z_j, but it makes no significant difference on our empirical results if the truncated range contains the interval [−8,8], for example, so from the practical viewpoint, we can ignore it.

	MC	NUS	LMS	NUS	LMS	NUS	LMS	NUS	NUS	NUS	NUS
s		1	2	2	3	3	5	5	10	20	100
ℓ_0	4.5	8.5	6.0	6.0	5.0	5.0	4.5	4.5	4.0	4.0	4.0
C	0.265	0.032	0.243	0.212	0.144	0.180	0.140	0.096	0.029	0.078	0.079
β	1.038	2.791	2.112	2.101	1.786	1.798	1.301	1.270	1.011	0.996	1.010
δ	1.134	3.004	3.196	3.196	3.383	3.357	2.295	2.303	1.811	1.421	1.463
R^2	0.999	0.999	1.000	1.000	0.995	0.995	0.979	0.978	0.990	0.991	0.996
$\hat{\kappa}_*$	1.121	0.925	1.238	1.215	1.156	1.191	1.109	1.045	0.820	0.925	0.934
$\hat{\gamma}_*$	0.202	0.398	0.293	0.292	0.242	0.244	0.207	0.201	0.174	0.184	0.185
ℓ_*	3.675	7.682	5.268	5.266	4.386	4.391	3.776	3.765	3.590	3.604	3.612
$\hat{K}_*$	0.299	0.071	0.221	0.205	0.163	0.184	0.173	0.137	0.061	0.117	0.119
$\hat{\nu}_*$	0.808	1.594	1.174	1.168	0.967	0.978	0.826	0.806	0.696	0.735	0.740
$\tilde{\nu}$	0.781	1.595	1.176	1.169	0.976	0.975	0.832	0.806	0.744	0.764	0.774
e19	17.01	34.06	24.39	24.38	20.79	20.80	17.88	17.79	17.28	17.07	17.05

Table 3.1. Parameter estimates for the KDE, for a sum of normals, over [−2,2].

We estimate the density over $[a,b] = [-b,b] = [-2,2]$. In our first experiment, we take $a_1 = \cdots = a_s = 1$, so all the coordinates have the same importance (which is disadvantageous for RQMC). Later, we will consider varying coefficients a_j. Table A.6 summarizes the results when B is estimated. For MC, our estimates given in the first column are based on experiments made with $s = 1$, but are valid for all s, because the IV and ISB do not depend on s. The estimated values for MC agree with the theory: the exact asymptotic values are $\gamma = 0.2$, $\nu = 0.8$, and $\beta = \delta = 1$. The other columns give some results for Sobol'+LMS and Sobol'+NUS, for selected values of s. For all $s > 1$ that we have tried, LMS and NUS give almost the same values. The first rows give the dimension s, the ℓ_0 found by pilot runs and used to fit the IV model, the estimated parameters C, β, and δ of the IV model, the

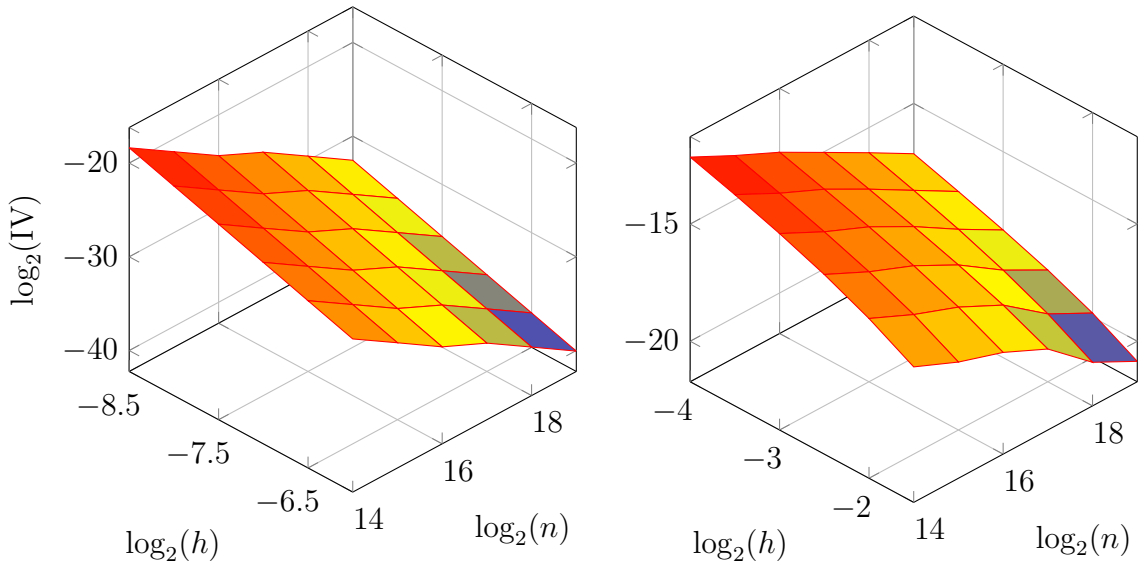

Figure 3.1. $\log_2(\text{IV})$ for the KDE with Sobol'+NUS for $s = 1$ (left) and $s = 20$ (right).

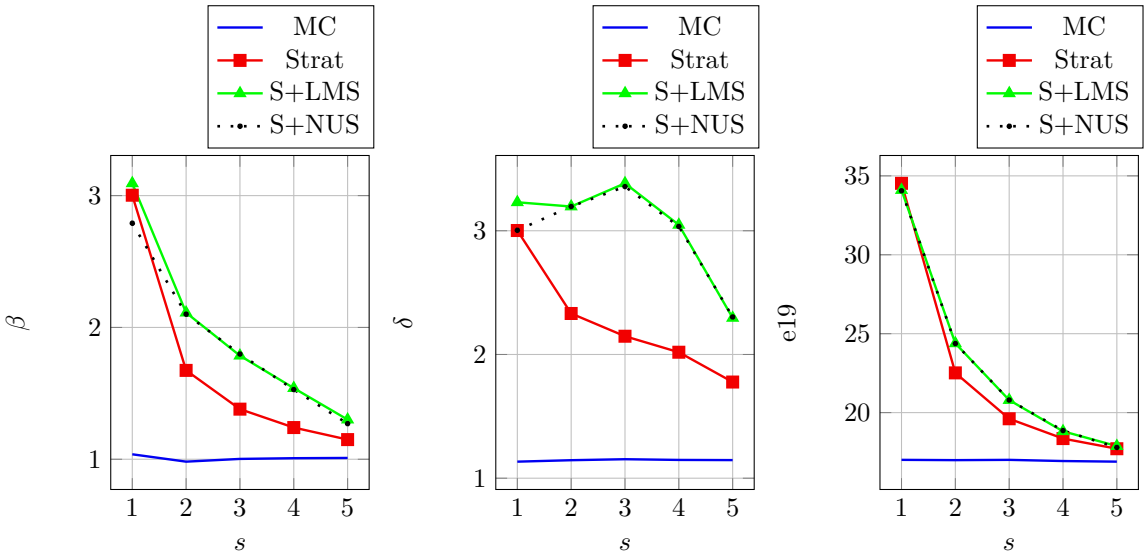

Figure 3.2. Estimated β, δ, and e19 with MC, Stratification, Sobol'+LMS, and Sobol'+NUS.

fraction R^2 of variance explained by this model, and the estimated B. The other quantities are defined in Section 3.6.1, except for $\ell_* = -\log_2 \hat{h}_*(2^{19})$, which gives an idea of the optimal h for $n = 2^{19}$.

Recall that the rates $\tilde{\nu}$ and e19 are obtained from a second-stage experiment, by using the estimated $\hat{h}_*(n)$ from the model in the first stage. All the R^2 coefficients are close to 1, which means that the log-log linear model is reasonably good in the area considered. The estimate of B is $B \approx 0.0418$ (same first three digits) for all s and all RQMC methods. Thus,

this estimator of B has very little variance. The MISE reduction of RQMC vs MC can be assessed by comparing their values of e19 given in the last row. For example, with the KDE for $s = 1$, the MISE for $n = 2^{19}$ is approximately 2^{-34} for Sobol'+NUS compared to 2^{-17} for MC, i.e., about $2^{17} \approx 125{,}000$ times smaller. For $s = 2$, for both LMS and NUS, the MISE is about $2^{-24.3}$, which is about 150 times smaller than for MC.

Figure 3.1 gives a visual assessment of the fit of the linear model for $\log_2(\text{IV})$ in the selected region, for two values of s. We made similar plots for several $s > 1$ and all point sets, and the linear approximation looked reasonable in all cases. Figure A.1 shows the estimated β, δ, and e19, for $s = 1, \ldots, 5$, for various point sets. Stratification, shown here and not in the table, is exactly equivalent to Sobol'+NUS for $s = 1$, and somewhat less effective for $s > 1$.

One important observation from the plots and the last row of the table (e19) is that for all s, the RQMC methods never have a larger MISE than MC. Their MISE is much smaller for small s, and becomes almost the same as for MC when s gets large. The MISE rate $\tilde{\nu}$ behaves similarly. Another important observation is that the coefficients β and δ in the IV model (which are both 1 with MC) are *both* larger than 1 with RQMC. For small s, with RQMC, β is significantly larger than $\tilde{\nu}$, which means that the IV converges must faster as a function of n when h is fixed than when h varies with n to optimize the MISE. This is explained by the large values of δ, sometimes even larger than 3, which indicate that reducing h to reduce the ISB increases the IV rapidly, and this limits the MISE reduction that we can achieve.

Here f is the standard normal density and $R(f'') = [-b\,(2b^2 - 1)\,e^{-b^2} + 3\int_0^b e^{-x^2}\,\mathrm{d}x]/4\pi$. For $b = 2$, this gives $R(f'') \approx 0.19018$, so the true constant B in the AISB is $B = R(f'')/4 \approx 0.04754$, whereas our estimate was 0.0418 for all s and all point sets. The difference is not due to noise, but is a bias coming from the fact that we estimated $R(f'')$ via KDE with finite n. We verified empirically that when we estimate these quantities with a larger n, the bias decreases slowly and appears to converge to 0 when $n \to \infty$.

We repeated the density estimation experiment by using the exact values of B instead of the estimated ones to choose h, and the results were very close for all s. In particular, the MISE rates $\tilde{\nu}$ and the values of e19 were almost the same.

We now take different coefficients (weights) a_j in the linear combination of the Z_j that defines X. Our purpose is to illustrate that there are situations where RQMC can perform very well with the KDE even when the dimension s is large. This can occur for example if the effective dimension is not large; i.e., when $g(\mathbf{u})$ depends mostly on just a few coordinates of $\mathbf{u}$, and does not vary much with respect to the other coordinates (Caflisch et al., 1997; L'Ecuyer, 2009). To illustrate this, we take $a_j = 2^{-j}$ for $j = 1, \ldots, s$, and we repeat the same set of experiments as we did for equal weights, to estimate the density over $[-2,2]$.

s	MC	2	4	10	20	50	100
C	0.171	0.173	0.038	6.7E-3	8.0E-3	7.3E-3	7.9E-3
β	1.000	2.100	1.650	1.420	1.427	1.425	1.429
δ	1.137	3.189	3.745	3.626	3.582	3.604	3.603
$\hat{K}_*$	0.213	0.183	0.080	0.032	0.035	0.033	0.035
$\hat{\nu}_*$	0.779	1.168	0.852	0.745	0.753	0.750	0.752
$\tilde{\nu}$	0.774	1.176	0.892	0.750	0.730	0.758	0.752
e19	16.96	24.76	19.71	18.96	18.98	18.99	19.04

Table 3.2. Parameter estimates for the KDE under Sobol'+LMS, for a weighted sum of normals with $a_j = 2^{-j}$.

Table 3.2 summarizes our findings for Sobol'+LMS, for s up to 100. The results with Sobol'+NUS are very similar. For $s = 1$, the results are obviously the same as for our previous setting, but they diverge when we increase s. For example, in the previous setting, the MISE estimate with $n = 2^{19}$ for $s = 2$, 10, and 100, was $2^{-24.38}$, $2^{-17.28}$, and $2^{-17.05}$, respectively, whereas with the new weights, it is $2^{-24.76}$, $2^{-18.96}$, and $2^{-19.04}$, respectively. For $s = 100$, in particular, the MISE with RQMC and $n = 2^{19}$ was about the same as for MC in the previous setting, and it is reduced by a factor of 4 in the present setting. We also see from the table that in 10 or more dimensions, the convergence rate of the MISE is not improved, but the constant is improved (empirically). As expected, when s increases beyond about 10, all the model parameters appear to stabilize as a function of s. In the previous setting, they were stabilizing around the MC values, but now they stabilize to different values. For example, in $s = 100$ dimensions, β was near the MC value of 1, and now it is about 1.4.

3.6.3. Displacement of a cantilevel beam

Bingham (2017) gives the following simple model of the displacement D of a cantilever beam with horizontal and vertical loads:

$$D = \frac{4L^3}{Ewt}\sqrt{\frac{Y^2}{t^4} + \frac{X^2}{w^4}} \tag{3.6.4}$$

in which L is the length of the beam, fixed to 100 inches, w and t are the width and thickness of the cross-section, taken as 4 and 2 inches, while X, Y, and E are assumed independent and normally distributed with means and standard deviations given as follows (in inches):

Description	Symbol	Mean	St. dev.
Young's modulus	E	2.9×10^7	1.45×10^6
Horizontal load	X	500	100
Vertical load	Y	1000	100

We want to estimate the density of the relative displacement $\tilde{X} = D/D_0 - 1$, where $D_0 = 2.2535$ inches. Here, the exact density is unknown, so unbiased estimators of the ISB and the MISE are not available, but we can estimate the AISB as in the previous example, and use it to estimate the optimal h and the MISE. A plot of the estimated density, obtained with a KDE with Sobol'+NUS and $n = 2^{19}$ points, is given in Figure 3.3. For the experiments reported here, we estimate the density of $\tilde{X}$ over the interval [0.407,1.515], which covers about 99% of the density (it excludes roughly 0.5 % on each side).

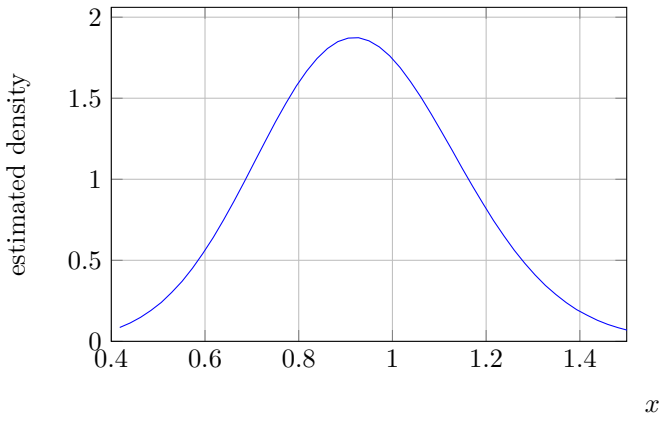

Figure 3.3. Estimated density of $\tilde{X}$, the relative displacement of a cantilever beam.

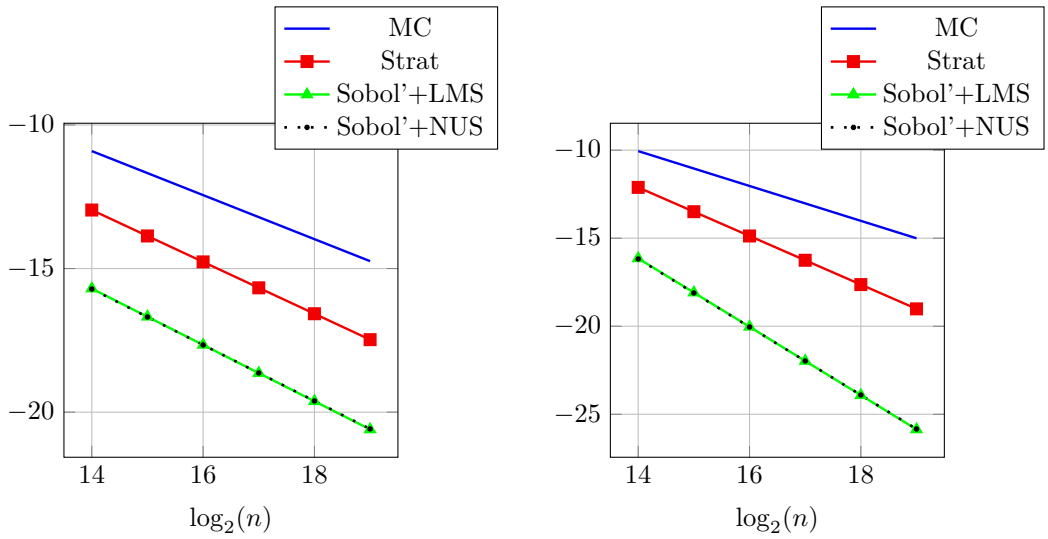

Figure 3.4. Estimated MISE (left) and IV (right) as a function of n for $h = 2^{-6}$, for the cantilever example.

	MC	Strat	LMS	NUS
C	0.109	0.022	1.8E-4	1.5E-4
β	0.991	1.380	1.943	1.932
δ	1.168	2.113	3.922	3.933
R^2	0.999	0.999	0.999	0.999
B	107.4	107.2	107.1	107.1
$\hat{\kappa}_*$	0.208	0.225	0.186	0.182
$\hat{\gamma}_*$	0.192	0.226	0.245	0.244
ℓ_*	5.909	6.443	7.090	7.085
$\hat{K}_*$	0.885	0.800	0.256	0.237
$\hat{\nu}_*$	0.767	0.903	0.981	0.974
e19	14.74	17.48	20.60	20.58

Table 3.3. Experimental results for the KDE, for the displacement of a cantilever beam, over the interval [0.407,1.515].

Table A.9 gives the parameter estimates from our experiment. RQMC increases the rate β from 1 to about 2. However, δ increases even more, from 1 to about 4. This means that although the variance decreases much faster than for MC as a function of n for fixed h, we cannot afford to decrease h very much to decrease the bias, so the MISE reduction is limited. RQMC improves both the estimated rate $\hat{\nu}_*$ and the constant K in the MISE model.

Figure A.2 shows the estimated MISE as a function of n (with the estimated optimal h), as well as the estimated IV as a function of n, all in log scale. The results for Sobol'+LMS and Sobol'+NUS are practically indistinguishable in those plots. We see that although the

MISE rate (slope) is not improved much by RQMC, the MISE is nevertheless reduced by a significant factor. With $n = 2^{19}$, the MISE is almost $2^6 = 64$ times smaller with Sobol'+LMS than with MC. For fixed h, the IV converges at a faster rate with RQMC than with MC.

Here, g is strictly decreasing in E and strictly increasing in both X and Y. Therefore, Corollary 3.5.1 applies. The asymptotic parameter values are $\beta = 4/3$, $\delta = 2$, and $\nu = 0.889$, which are very close to what we found empirically for stratification (see A.9).

3.6.4. A weighted sum of lognormals

In this example, we estimate the density of a weighted sum of lognormals: $X = \sum_{j=1}^{s} w_j \exp(Y_j)$ where $\mathbf{Y} = (Y_1, \ldots, Y_s)^\mathsf{t}$ has a multinormal distribution with mean vector μ and covariance matrix $\mathbf{C}$. Let $\mathbf{C} = \mathbf{A}\mathbf{A}^\mathsf{t}$ be a decomposition of $\mathbf{C}$. To generate $\mathbf{Y}$, we generate $\mathbf{Z}$ a vector of s independent standard normals by inversion, then put $\mathbf{Y} = \mu + \mathbf{A}\mathbf{Z}$. For MC, the choice of decomposition does not matter, but for RQMC it does, and here we take the decomposition used in principal component analysis (PCA) (Glasserman, 2004; L'Ecuyer, 2009). We also tried sequential sampling (SS) and Brownian bridge sampling (BBS) but with them, RQMC did not improve the IV significantly as we will see with PCA.

This model has several applications. In one of them, for some positive constants ρ and s_0, by taking $w_j = s_0(s-j+1)/s$, $e^{-\rho}\max(X-K,0)$ is the payoff of a financial option based on the average value of a stock or commodity price at s observation times, under a geometric Brownian motion process. Estimating the density of this random payoff in its positive part is equivalent to estimating the density of X over the interval (K,∞) (for simplicity we ignore the scaling factor $e^{-\rho}$). When we compute the KDE here, the realizations of X smaller than K are *not* discarded; they contribute to the KDE slightly above K. Discarding them would introduce a significant bias in the KDE due to a boundary effect at K.

For our numerical experiment, we take this special case with the same parameters as in L'Ecuyer (2018): $s = 12$, $s_0 = 100$, and $K = 101$. The matrix $\mathbf{C}$ is defined indirectly as follows. We have $Y_j = Y_{j-1}(\mu - \sigma^2)j/s + \sigma B(j/s)$ where $Y_0 = 0$, $\sigma = 0.12136$, $\mu = 0.1$, and $B(\cdot)$ is a standard Brownian motion. We estimate the density of $\tilde{X} = X - K$ over the interval $[a,b] = [0, 27.13]$. Approximately 0.5% of the density lies on the right of this interval and 29.05% lies on the left (this is when the option brings no payoff). Figure 3.5 shows a

plot of the estimated density of $\tilde{X} = X - K$ obtained from a KDE with Sobol'+NUS and $n = 2^{19}$ points.

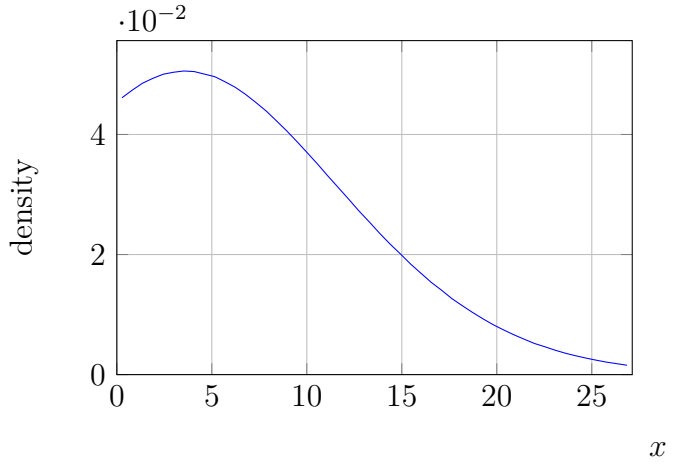

Figure 3.5. Estimated density of the option payoff $X - K$.

	MC	LMS	NUS
C	0.171	0.110	0.097
β	1.005	1.671	1.663
δ	1.151	4.907	4.930
R^2	0.999	0.990	0.990
B	1.1E-6	1.1E-6	1.1E-6
$\hat{\kappa}_*$	7.953	3.717	3.657
$\hat{\gamma}_*$	0.195	0.188	0.186
ℓ_*	0.715	1.670	1.668
$\hat{K}_*$	0.020	3.9E-4	3.6E-4
$\hat{\nu}_*$	0.780	0.750	0.745
e19	20.45	25.59	25.58

Table 3.4. Experimental results for the density estimation of the option payoff over the interval $[0, 27.13]$.

Table A.11 summarizes the results of our experiments. Again, the linear model for the IV fits extremely well in the selected area. RQMC improves β from 1 to about 5/3, which is significant, but at the same time δ increases from about 1.1 to nearly 5. This means we are very limited in how much we can decrease h to reduce the bias. On the other hand, this empirical δ is not as bad as the one in the AIV bound of Corollary 3.4.1, which gives $\delta = 2s = 24$. The estimate of B is again about the same for all point sets. Somewhat surprisingly, in the region considered, the estimated MISE rate $\hat{\nu}_*$ is not better for RQMC than for MC, due to the large δ, but the MISE is nevertheless about 32 times smaller for

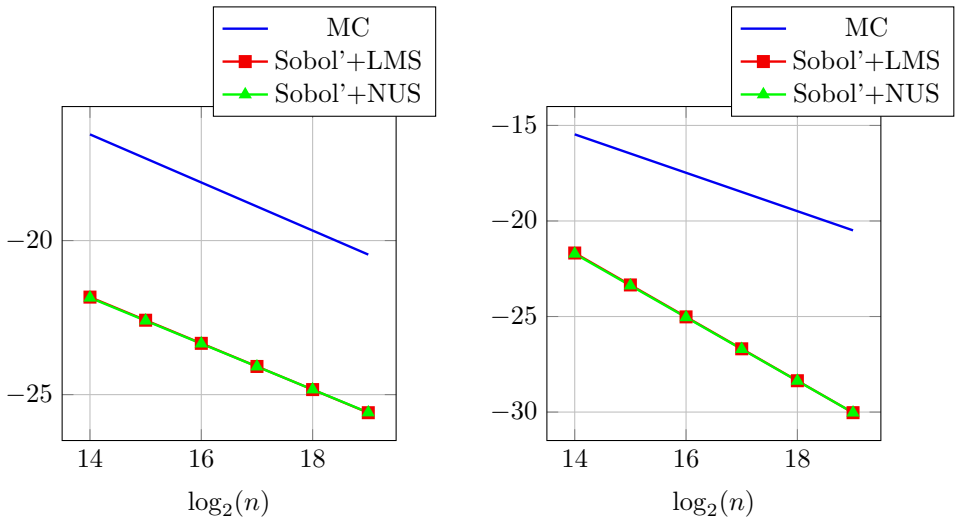

Figure 3.6. Estimated MISE as a function of n (left) and estimated IV as a function of n for $h = 1/2$ (right).

RQMC than for MC in the range of interest, as shown in Figure A.3, for which h was taken as the estimated optimal h from our model, as a function of n. That is, RQMC is truly beneficial for estimating the payoff density in this 12-dimensional example. In the lower panel, we see that the estimated IV for fixed h converges faster with RQMC than with MC.

3.7. Conclusion

We explored RQMC combined with KDEs to estimate a density by simulation. RQMC can improve the IV and the MISE, sometimes by large factors, in situations in which the (effective) dimension is small. The improvement is more limited when the effective dimension is large. We also found that the IV improvement degrades quickly as a function of h when $h \to 0$. In our empirical experiments, the IV was never larger with KDE+RQMC than with KDE+MC, and was often much smaller.

In the online supplement, we report a similar analysis for histograms instead of KDEs, and find that RQMC also brings some improvement, but more limited than with the KDE. We also provide additional experimental results and details.

Acknowledgments

The idea of this work started during a workshop at the Banff International Research Station (BIRS) in October 2015. Most of the research was accomplished within a research

program on quasi-Monte Carlo sampling methods at the Statistical and Applied Mathematical Sciences Institute (SAMSI), in North Carolina, in 2017–2018.

Chapter 4

Article 2: Monte Carlo and Quasi-Monte Carlo Density Estimation via Conditioning

In the second article, we propose a conditional Monte Carlo (CMC) method for estimating the density of a continuous random variable when its random realizations are generated from a simulation model. The resulting estimator, called the conditional density estimator (CDE), is unbiased and has the canonical $\mathcal{O}(n^{-1})$ convergence rate in its MISE, which improves upon the popular KDE method. In addition, we show that combining CDE with RQMC achieve a even faster $\mathcal{O}(n^{-2+\epsilon})$ rate for the MISE. We compare the proposed CDE (CMC+RQMC) with the KDE (KDE+RQMC) and a recently proposed gradient estimation method (with RQMC), the generalized likelihood ratio (GLR) method, via plenty of numerical examples including realistic examples. All the numerical results indicate that the proposed CDE+RQMC method has a small MISE and outperforms others.

This article is currently under revision in INFORMS Journal on Computing. Preliminary work was presented at the following conferences:

— The Optimization Days, Montreal, May 2019;

— "Meet a GERAD researcher!" Seminar, Montreal, March 2019.

The main author contributions are:

— The general research ideas were proposed by Pierre L'Ecuyer.

— The implementations and experiments were carried out by Amal Ben Abdellah with the help of Florian Puchhammer.

— All authors contributed to the writing, although P. L'Ecuyer wrote the final version.

Monte Carlo and Quasi-Monte Carlo Density Estimation via Conditioning

Pierre L'Ecuyer

Département d'Informatique et de Recherche Opérationnelle, Pavillon Aisenstadt, Université de Montréal, C.P. 6128, Succ. Centre-Ville, Montréal, Québec, Canada H3C 3J7. lecuyer@iro.umontreal.ca

Florian Puchhammer

Basque Center for Applied Mathematics, Alameda de Mazarredo 14, 48009 Bilbao, Basque Country, Spain; and Département d'Informatique et de Recherche Opérationnelle, Université de Montréal, fpuchhammer@bcamath.org

Amal Ben Abdellah

Département d'Informatique et de Recherche Opérationnelle, Pavillon Aisenstadt, Université de Montréal, C.P. 6128, Succ. Centre-Ville, Montréal, Québec, Canada H3C 3J7, amal.ben.abdellah@umontreal.ca

Abstract

Estimating the unknown density from which a given independent sample originates is more difficult than estimating the mean, in the sense that for the best popular density estimators, the mean integrated square error converges slower than at the canonical rate of $\mathcal{O}(1/n)$. When the sample is generated from a simulation model and we have control over how this is done, we can do better. We examine an approach in which conditional Monte Carlo permits one to obtain a smooth estimator of the cumulative distribution function, whose sample derivative is an unbiased estimator of the density at any point, and therefore converges at a faster rate than the usual density estimators. We can achieve an even faster rate by combining this with randomized quasi-Monte Carlo to generate the samples.

Key words: density estimation; conditional Monte Carlo; quasi-Monte Carlo.

4.1. Introduction

Simulation is commonly used to generate n realizations of a random variable X that may represent a payoff, a cost, or a performance of some kind, and then to estimate from this sample the unknown expectation of X together with a confidence interval (Asmussen and

Glynn, 2007; Law, 2014). Simulation books focus primarily on how to improve the quality of the estimator of $\mathbb{E}[X]$ and of the confidence interval. Estimating a given quantile of the distribution of X, or the sensitivity of $\mathbb{E}[X]$ with respect to some parameter in the model, for example, are also well studied topics.

However, large simulation experiments can provide a lot more information than just point estimates with confidence intervals. Running simulations of a complex system for hours, with thousands on runs, only to report confidence intervals on a few single numbers is poor data valorization. Simulation experiments can give much more useful information than that. In particular, it can provide an estimate of the entire distribution of X, and not only its expectation or a specific quantile. Leading simulation software routinely provide histograms that give a rough idea of the distribution of the output random variables of interest, and users certainly appreciate this type of visual display. When X has a continuous distribution, a histogram is just a primitive form of density estimator. Offering a more accurate estimator of the entire distribution is at least as important and useful asgiving a more accurate confidence interval on the mean.

As an illustration, when simulating a large call center with several different call types, one can compute and report a confidence interval on the expected waiting time for each call type, or perhaps on the probability that a call waits more than 30 seconds. But from the same simulations, one can provide an estimate of the entire waiting time distribution for each call type. As another example, for a large project made of several activities with random durations, with precedence relationships between certain activities, one can simulate n realizations of the model and compute a confidence interval on the expected total duration of the project. But from the same simulations, one can estimate the whole distribution of the (random) project duration, and this is much more useful.

One way to visualize the *entire distribution of* X is to look at the empirical *cumulative distribution function* (cdf) of the observations. But density estimators (including histograms) are preferred because they give a better visual insight on the distribution than the cdf. However, accurate density estimation is generally hard. Given n independent realizations of X, the mean integrated square error (MISE) between the true density and a histogram with optimally selected divisions converges only as $\mathcal{O}(n^{-2/3})$. There are more refined methods than histograms, the leading one being the *kernel density estimator* (KDE) whose MISE converges

as $\mathcal{O}(n^{-4/5})$ in the best case. These rates are slower than the canonical $\mathcal{O}(n^{-1})$ rate for the variance of the sample average as an unbiased estimator of the mean. The slower rates stem from the presence of bias. For a histogram, taking wider rectangles reduces the variance but increases the bias by flattening out the short-range density variations. A compromise must be made to minimize the MISE. The same happens with the KDE, with the rectangle width replaced by the bandwidth of the kernel. Selecting a good bandwidth for the KDE is particularly difficult. The bandwidth should ideally vary over the interval in which we estimate the density; it should be smaller where the density is larger and/or smoother, and vice-versa. This is complicated to implement. Handling discontinuities in the density is also problematic. These difficulties have discouraged the use of KDEs in reporting simulation results.

The KDE and other related density estimation methods where developed mainly for the situation were n independent realizations of X are given in advance and nothing else is known, as traditionally assumed in classical non-parametric statistics, and one wishes to estimate the density from them (Scott, 2015). But in a Monte Carlo setting in which the n observations are generated by simulation, there are opportunities to do better by controlling the way we generate the realizations and by exploiting the fact that we know the underlying stochastic model. This is the subject of the present paper.

Our approach combines two general methods. The first general idea is to build a smooth estimator of the cdf via conditional Monte Carlo (CMC), and take the sample derivative of this estimator to estimate the density. We call it a *conditional density estimator* (CDE). Under appropriate conditions, the CDE is unbiased and its variance is bounded uniformly by a constant divided by n, so its MISE is $\mathcal{O}(n^{-1})$. This idea of using CMC was mentioned by Asmussen and Glynn (2007), page 146, Example 4.3, and further studied in Asmussen (2018), both for the special case where the goal is to estimate the density of a sum of continuous random variables having a known distribution from which we can sample exactly. Asmussen (2018) simply "hides" the last term of the sum, meaning that this term is not generated, and he takes the conditional distribution of the sum given the other terms, to estimate the cdf, the density, the value at risk, and the conditional value at risk of the sum. Smoothing by CMC before taking a stochastic derivative has been studied long ago for estimating the derivative of an expectation (Fu and Hu, 1997; Gong and Ho, 1987; L'Ecuyer and Perron,

1994) and the derivative of a quantile (Fu et al., 2009) with respect to a model parameter. This is known as *smoothed perturbation analysis*. However, no body noticed that this could be used for density estimation until Asmussen used it for his special case.

One of our main goals in this paper is to show that this CDE method can be used to estimate the density in a much more general setting than Asmussen (2018), and to examine how effective it is via experiments on several types of examples. In most of these examples, X is *not* defined as a sum of random variables, and we often have to hide more than just one random variableto do the conditioning. We give conditions under which we can prove that the density estimator is unbiased. A key condition is that the conditional cdf must be a continuous function of the point x at which we estimate the density. The variance of the density estimator may depend strongly on which variables we hide, i.e., on what we are conditioning. We illustrate this with several examples and we provide guidelines for the choice of conditioning.

Once we have a smooth density estimator, the second (complementary) strategy to further reduce the MISE and even improve its convergence rate is to replace the independent uniform random numbers that drive the simulation by *randomized quasi-Monte Carlo* (RQMC) points. We show in this paper that by combining these two strategies, under appropriate conditions, we can obtain a density estimator whose MISE converges at a faster rate than $\mathcal{O}(n^{-1})$, for instance $\mathcal{O}(n^{-2+\epsilon})$ for any $\epsilon > 0$ in some situations. We also observe this fast rate empirically on numerical examples. This happens essentially when the CDE is a smooth function of the underlying uniforms. To our knowledge, this type of convergence rate has never been proved or observed before for density estimation. The combination of RQMC with an ordinary KDE was studied by Ben Abdellah et al. (2019a), who were able to prove a faster rate than $\mathcal{O}(n^{-4/5})$ for the MISE when the RQMC points have a small number of dimensions. They observed this faster rate empirically on examples. They also showed that the MISE reduction provided by RQMC degrades rapidly when the bandwidth is reduced (to reduce the bias) or when the dimension increases. The CDE+RQMC approach studied in the present paper avoids this problem (there is no bias and no bandwidth) and is generally more effective than the KDE+RQMC combination. We provide some numerical comparisons in our examples.

Other Monte Carlo density estimators were proposed recently, based on the idea of estimating the derivative of the cdf using a likelihood ratio (LR) method. This general approach permits one to estimate the derivative of the expectation of a random variable with respect to some parameter of the underlying distribution when this random variable is discontinuous (Glynn, 1987; L'Ecuyer, 1990). Laub et al. (2019) proposed an approach that combines a clever change of variable with the LR method to estimate the density of a sum of random variables as in Asmussen (2018), but in a setting where the random variables are dependent. Peng et al. (2018) proposed a generalized version of the LR gradient estimator (GLR) to estimate the derivative of an expectation with respect to a more general model parameter. Lei et al. (2018) then sketched out how GLR could be used to estimate a density. Formulas for these GLR density estimators are given in Theorem 1 of Peng et al. (2020). A referee pointed out these last two papers to us. We compare our method with these GLR-based estimators in our numerical illustrations.

Density estimation has other applications than just visualizing the distribution of an output random variable (Scott, 2015; Van der Vaart, 2000). For instance when computing a confidence interval for a quantile using the central-limit theorem (CLT), one needs a density estimator at the quantile to estimate the variance (Asmussen and Glynn, 2007; Nakayama, 2014a,b; Peng et al., 2017; Serfling, 1980). Another application is for maximum likelihood estimation when the likelihood does not have a closed-form expression, so to maximize it with respect to some parameter θ, the likelihood function (which in the continuous case is a density at any value of θ) must be estimated (Peng et al., 2020; Van der Vaart, 2000).

A common setting is when a set of observations is given and one wishes to estimate the density from which they come. A different one is when we have a simulation (or generative) model from which observations can be generated and we want to estimate the density of the model output. This is the focus of our paper. Density estimates are also important in various other settings. For instance when computing a confidence interval for a quantile using the central-limit theorem (CLT), one needs a density estimator at the quantile to estimate the variance (Asmussen and Glynn, 2007; Nakayama, 2014a,b; Peng et al., 2017; Serfling, 1980). Another application is for maximum likelihood estimation when the likelihood does not have a closed-form expression, so to maximize it with respect to some parameter θ, the likelihood function (which in the continuous case is a density at any value of θ) must be estimated

(Peng et al., 2020; Van der Vaart, 2000). A related application is the estimation of the posterior density of θ given some data, in a Bayesian model (Efron and Hastie, 2016).

The remainder is organized as follows. In Section 4.2, we define our general setting, recall some key facts about density estimators, introduce the general CMC method to build the CDEs considered in this paper, prove some of their properties, and give small examples to provide insight on the key ideas. In Section 4.3, we explain how to combine the CDE with RQMC and discuss the convergence properties for this combination. Section 6.4 reports experimental results with various examples. Some of the examples feature creative ways of conditioning to boost the effectiveness of the method. A conclusion is given in Section 5.6. The main ideas of this paper were presented at a SAMSI workshop on QMC methods in North Carolina, and at a RICAM workshop in Linz, Austria, both in 2018.

4.2. Model and conditional density estimator

4.2.1. Density estimation

We have a real-valued random variable X that can be simulated from its exact distribution, but we do not know the cdf F and density f of X. Typically, X will be an easily computable function of several other random variables with known densities. Our goal is to estimate f over a finite interval $[a,b]$. Let $\hat{f}_n$ denote an estimator of f based on a sample of size n. We measure the quality of $\hat{f}_n$ by the *mean integrated square error* (MISE), defined as

$$\mathrm{MISE} = \mathrm{MISE}(\hat{f}_n) = \int_a^b \mathbb{E}[\hat{f}_n(x) - f(x)]^2 \mathrm{d}x. \tag{4.2.1}$$

The MISE is the sum of the *integrated variance* (IV) and the *integrated square bias* (ISB):

$$\mathrm{MISE} = \mathrm{IV} + \mathrm{ISB} = \int_a^b \mathbb{E}(\hat{f}_n(x) - \mathbb{E}[\hat{f}_n(x)])^2 \mathrm{d}x + \int_a^b (\mathbb{E}[\hat{f}_n(x)] - f(x))^2 \mathrm{d}x.$$

A standard way of constructing $\hat{f}_n$ when $X_1, \ldots, X_n$ are n independent realizations of X is via a KDE, defined as follows (Parzen, 1962; Scott, 2015):

$$\hat{f}_n(x) = \frac{1}{nh} \sum_{i=1}^{n} k\left(\frac{x - X_i}{h}\right),$$

where the *kernel* k is a probability density over $\mathbb{R}$, usually symmetric about 0 and non-increasing over $[0,\infty)$, and the constant $h > 0$ is the *bandwidth*, whose role is to stretch [or

compress] the kernel horizontally to smooth out [or unsmooth] the estimator $\hat{f}_n$. The KDE was developed initially for the setting in which $X_1, \ldots, X_n$ are given a priori, and it is still the most popular one for this situation. It can be used in exactly the same way when $X_1, \ldots, X_n$ are independent observations produced by simulation from a generative model, but in that case there is an opportunity to do better, as we now explain.

4.2.2. Conditioning and the stochastic derivative as an unbiased density estimator

Since the density of X is the derivative of its cdf, $f(x) = F'(x)$, a natural idea would be to take the derivative of an estimator of the cdf as a density estimator. The simplest candidate for a cdf estimator is the *empirical cdf*

$$\hat{F}_n(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}[X_i \le x],$$

but $\mathrm{d}\hat{F}_n(x)/\mathrm{d}x = 0$ almost everywhere, so this one *cannot* be a useful density estimator. Here, $\hat{F}_n(x)$ is an unbiased estimator of $F(x)$ at each x, but its derivative is a biased estimator of $F'(x)$. That is, because of the discontinuity of $\hat{F}_n$, we cannot exchange the derivative and expectation:

$$0 \;=\; \mathbb{E}\left[\frac{\mathrm{d}\hat{F}_n(x)}{\mathrm{d}x}\right] \;\neq\; \frac{\mathrm{d}\mathbb{E}[\hat{F}_n(x)]}{\mathrm{d}x} \;=\; F'(x).$$

A continuous estimator of F can be constructed by CMC, as follows. Replace the indicator $\mathbb{I}[X \le x]$ by its *conditional cdf* given filtered (reduced) information $\mathcal{G}$: $F(x \mid \mathcal{G}) \stackrel{\text{def}}{=} \mathbb{P}[X \le x \mid \mathcal{G}]$, where $\mathcal{G}$ is a sigma-field that contains not enough information to reveal X but enough to compute $F(x \mid \mathcal{G})$. Here, knowing the realization of $\mathcal{G}$ means knowing the realizations of all $\mathcal{G}$-measurable random variables. Our CDE to estimate $f(x)$ will be the *conditional density* $f(x|\mathcal{G}) \stackrel{\text{def}}{=} F'(x|\mathcal{G}) = \mathrm{d}F(x \mid \mathcal{G})/\mathrm{d}x$, if it exists. Under the following assumption, we prove that $f(x|\mathcal{G})$ is an unbiased estimator of $f(x)$ whose variance is bounded uniformly in x. Note that since $F(\cdot \mid \mathcal{G})$ cannot decrease, $f(\cdot \mid \mathcal{G})$ is never negative.

Assumption 4.2.1. *For all realizations of $\mathcal{G}$, $F(x \mid \mathcal{G})$ is a continuous function of x over the interval $[a,b]$, and is differentiable except perhaps at a countable set of points $D(\mathcal{G}) \subset [a,b]$. There is also a random variable Γ defined over the same probability space as $F(x \mid \mathcal{G})$, such that $\mathbb{E}[\Gamma^2] \le K_\gamma$ for some constant $K_\gamma < \infty$, and for which $\sup_{x \in [a,b] \setminus D(\mathcal{G})} F'(x \mid \mathcal{G}) \le \Gamma$.*

Proposition 4.2.1. *Under Assumption 4.2.1, we have* $\mathbb{E}[f(x \mid \mathcal{G})] = f(x)$ *and* $\mathrm{Var}[f(x \mid \mathcal{G})] \le K_\gamma$ *for all* $x \in [a,b]$.

Proof. We adapt the proof of Theorem 1 of L'Ecuyer (1990). By the mean value inequality theorem of Dieudonné (1969), Theorem 8.5.3, which is a form of mean value theorem for non-differentiable functions, for every $x \in [a,b]$ and $\delta > 0$, with probability 1, we have

$$0 \;\le\; \frac{\Delta(x,\delta,\mathcal{G})}{\delta} \;\stackrel{\text{def}}{=}\; \frac{F(x+\delta \mid \mathcal{G}) - F(x \mid \mathcal{G})}{\delta} \;\le\; \sup_{y \in [x,x+\delta] \setminus D(\mathcal{G})} F'(y \mid \mathcal{G}) \;\le\; \Gamma.$$

Then, by the dominated convergence theorem,

$$\mathbb{E}\left[\lim_{\delta \to 0} \frac{\Delta(x,\delta,\mathcal{G})}{\delta}\right] = \lim_{\delta \to 0} \mathbb{E}\left[\frac{\Delta(x,\delta,\mathcal{G})}{\delta}\right],$$

which shows the unbiasedness. Moreover, $\mathrm{Var}[F'(x \mid \mathcal{G})] \le \mathbb{E}[\Gamma^2] \le K_\gamma$. □

Suppose now that $\mathcal{G}^{(1)}, \ldots, \mathcal{G}^{(n)}$ are n independent realizations of $\mathcal{G}$, so $F(x \mid \mathcal{G}^{(1)}), \ldots, F(x \mid \mathcal{G}^{(n)})$ are independent realizations of $F(x \mid \mathcal{G})$, and consider the CDE

$$\hat{f}_{\text{cde},n}(x) = \frac{1}{n} \sum_{i=1}^{n} F'(x \mid \mathcal{G}^{(i)}). \tag{4.2.2}$$

It follows from Theorem 1 that $\mathrm{ISB}(\hat{f}_{\text{cde},n}) = 0$ and $\mathrm{MISE}(\hat{f}_{\text{cde},n}) = \mathrm{IV}(\hat{f}_{\text{cde},n}) \le (b-a)K_\gamma/n$. An unbiased estimator of this IV is given by

$$\widehat{\mathrm{IV}} = \widehat{\mathrm{IV}}(\hat{f}_{\text{cde},n}) = \frac{1}{n-1} \int_a^b \sum_{i=1}^{n} \left[F'(x \mid \mathcal{G}^{(i)}) - \hat{f}_{\text{cde},n}(x)\right]^2 \mathrm{d}x. \tag{4.2.3}$$

In practice, this integral can be approximated by evaluating the integrand at a finite number of points over $[a,b]$ and taking the average, multiplied by $(b-a)$.

It is well known that in general, when estimating $\mathbb{E}[X]$, a CMC estimator never has a larger variance than X itself, and the more information we hide, the smaller the variance. That is, if $\mathcal{G} \subset \tilde{\mathcal{G}}$ are two sigma-fields such that $\mathcal{G}$ contains only a subset of the information of $\tilde{\mathcal{G}}$, then

$$\mathrm{Var}[\mathbb{E}[X \mid \mathcal{G}]] \le \mathrm{Var}[\mathbb{E}[X \mid \tilde{\mathcal{G}}]] \le \mathrm{Var}[X]. \tag{4.2.4}$$

Noting that $F(x \mid \mathcal{G}) = \mathbb{E}[\mathbb{I}[X \le x] \mid \mathcal{G}]$, we also have

$$\mathrm{Var}[F(x \mid \mathcal{G})] \le \mathrm{Var}[F(x \mid \tilde{\mathcal{G}})] \le \mathrm{Var}[\mathbb{I}[X \le x]] = F(x)(1-F(x)).$$

Thus, (4.2.4) applies as well to the (conditional) cdf estimator.

However, applying it to the CDE, which is the derivative of the conditional cdf, is less straightforward. It is obviously not true that $\mathrm{Var}[F'(x \mid \mathcal{G})] \le \mathrm{Var}[\mathrm{d}\mathbb{I}[X \le x]/\mathrm{d}x]$ because the latter is zero almost everywhere. Nevertheless, we can prove the following.

Lemma 4.2.1. *If $\mathcal{G} \subset \tilde{\mathcal{G}}$ both satisfy Assumption 4.2.1, then for all $x \in [a,b]$, we have* $\mathrm{Var}[f(x \mid \mathcal{G})] \le \mathrm{Var}[f(x \mid \tilde{\mathcal{G}})]$.

Proof. The result does not follow directly from (4.2.4) because F' is not an expectation; this is why our proof does a little detour. For an arbitrary $x \in [a,b]$ and a small $\delta > 0$, define the random variable $I = I(x,\delta) = \mathbb{I}[x < X \le x+\delta]$. We have $\mathbb{E}[I \mid \mathcal{G}] = F(x+\delta \mid \mathcal{G}) - F(x \mid \mathcal{G})$, as in the proof of Theorem 4.2.1, and similarly for $\tilde{\mathcal{G}}$. Using (4.2.4) with I in place of X gives

$$\mathrm{Var}[\mathbb{E}[I \mid \mathcal{G}]] \le \mathrm{Var}[\mathbb{E}[I \mid \tilde{\mathcal{G}}]]. \tag{4.2.5}$$

We have

$$f(x \mid \mathcal{G}) = \lim_{\delta \to 0} \frac{F(x+\delta \mid \mathcal{G}) - F(x \mid \mathcal{G})}{\delta} = \lim_{\delta \to 0} \mathbb{E}[I(x,\delta)/\delta \mid \mathcal{G}]$$

and similarly for $\tilde{\mathcal{G}}$. Combining this with (4.2.5), we obtain

$$\begin{aligned} \mathrm{Var}[f(x \mid \mathcal{G})] &= \mathrm{Var}[\lim_{\delta \to 0} \mathbb{E}[I(x,\delta)/\delta \mid \mathcal{G}]] = \lim_{\delta \to 0} \mathrm{Var}[\mathbb{E}[I(x,\delta)/\delta \mid \mathcal{G}]] \\ &\le \lim_{\delta \to 0} \mathrm{Var}[\mathbb{E}[I(x,\delta)/\delta \mid \tilde{\mathcal{G}}]] = \mathrm{Var}[\lim_{\delta \to 0} \mathbb{E}[I(x,\delta)/\delta \mid \tilde{\mathcal{G}}]] = \mathrm{Var}[f(x \mid \tilde{\mathcal{G}})], \end{aligned}$$

in which the exchange of "Var" with the limit (at two places) can be justified by a similar argument as in Proposition 4.2.1. More specifically, we need to apply the dominated convergence theorem to $\mathbb{E}[I(x,\delta)/\delta \mid \mathcal{G}]$, which is just the same as in Proposition 4.2.1, and also to its square, which is also valid because the square is bounded uniformly by Γ^2. This completes the proof. □

This lemma tells us that conditioning on less information (hiding more) always reduces the variance of the CDE (or keep it the same). But if we hide more, the CDE may be harder or more costly to compute, so a compromise must be made to minimize the work-normalized MISE (which is the MISE multiplied by the expected time to compute the estimator), and the best compromise is generally problem-dependent. When none of $\mathcal{G}$ or $\tilde{\mathcal{G}}$ is a subset of the other, the variances of the corresponding conditional density estimators may differ significantly, and Lemma 4.2.1 does not apply, so other guidelines must be used to select $\mathcal{G}$ when there are multiple possibilities.

In our setting, the most important condition is that $\mathcal{G}$ must satisfy Assumption 4.2.1. Any such $\mathcal{G}$ provides an unbiased density estimator with finite variance. When there are multiple choices, in general we want to choose $\mathcal{G}$ so that the conditional density tends to be spread out as opposed to being concentrated in a narrow peak. We give concrete examples of this in Section 6.4. We recognize that this criterion is heuristic. If f is very spiky itself, then the CDE must be spiky as well, because $\mathrm{Var}[X|\mathcal{G}] \leq \mathrm{Var}[X]$, and yet $\mathrm{Var}[f(x|\mathcal{G})]$ can be very small, even zero in degenerate cases. Also, a large $\mathrm{Var}[X|\mathcal{G}]$ for all $\mathcal{G}$ is not sufficient, because the large variance may come from two or more separate spikes, and this is why we write "spread out" instead of "large variance". Roughly, we want the CDE $f(.|\mathcal{G})$ to be spread out relative to f, for all $\mathcal{G}$.

A more elaborate selection criterion should take into account the IV of the CDE, its computing cost, and also some measure of smoothness of the resulting CDE as a function of the uniform random numbers, because this has an impact on RQMC effectiveness. For real-life models, it is usually much too hard to precompute such measures, so the best practice would be to identify a few promising candidates and either: (1) perform pilot runs to compare their effectiveness and select one or (2) take a convex combination of the corresponding CDEs, as explained in Section 4.2.4. We believe that finding a good $\mathcal{G}$ will always remain largely problem-dependent and it sometimes requires creativity. We illustrate this with a variety of examples in Section 6.4.

4.2.3. Small examples to provide insight

To illustrate some key ideas, this subsection provides simple examples to illustrate the key ideas. For these examples, we take $X = h(Y_1, \ldots, Y_d)$ where $Y_1, \ldots, Y_d$ are independent continuous random variables, each Y_j has cdf F_j and density f_j, and we condition on $\mathcal{G} = \mathcal{G}_{-k}$ defined as the information that remains after erasing the value taken by Y_k. We can write $\mathcal{G}_{-k} = (Y_1, \ldots, Y_{k-1}, Y_{k+1}, \ldots, Y_d)$. The CDE $f(x \mid \mathcal{G}_{-k})$ will be related to the density f_k and will depend on the form of h. In Section 6.4, we consider more elaborate forms of conditioning.

Example 4.2.1. A very simple situation is when $X = h(Y_1, \ldots, Y_d) = Y_1 + \cdots + Y_d$, a sum of d independent continuous random variables. By hiding Y_k for an arbitrary k, we get

$$F(x \mid \mathcal{G}_{-k}) \;=\; \mathbb{P}[X \leq x \mid S_{-k}] = \mathbb{P}[Y_k \leq x - S_{-k}] = F_k(x - S_{-k}),$$

where $S_{-k} \stackrel{\text{def}}{=} \sum_{j=1, j\neq k}^{d} Y_j$, and the density estimator becomes $f(x \mid \mathcal{G}_{-k}) = f_k(x - S_{-k})$. This form also works if the Y_j are not independent, provided that we are able to compute the density of Y_k conditional on $\mathcal{G}_{-k}$. It suffices to replace f_k by this conditional density. The model studied by Asmussen (2018) was a special case of this example, with independent variables and $k = d$.

When the Y_j's have different distributions and we want to hide one, which one should we hide? Intuition may suggest that we should always hide the one having the largest variance. Although this simple rule may work fine in a majority of cases, it is not always optimal. In particular, the optimal choice of k may depend on the value of x at which we estimate the density. To illustrate this, let $d = 2$, $f_1(y) = 2y$, and $f_2(y) = 2(1 - y)$, for $y \in (0,1)$. Then, $f(x) > 0$ for $0 < x < 2$. If we hide Y_2, the density estimator at x is $f_2(x - Y_1)$ and its second moment is $\mathbb{E}[f_2^2(x - Y_1)] = \int_0^1 f_2^2(x - y_1) f_1(y_1) \mathrm{d}y_1$ whereas if we hide Y_1 instead, the density estimator at x is $f_1(x - Y_2)$ and its second moment is $\mathbb{E}[f_1^2(x - Y_2)] = \int_0^1 f_1^2(x - y_2) f_2(y_2) \mathrm{d}y_2$. One can easily verify that when x is close to 0, these integrands are positive only when both y_1 and y_2 are also close to 0, and then the second integral is smallest, so it is better to hide Y_1. When x is close to 2, the opposite is true and it is better to hide Y_2.

In applications, changing the conditioning as a function of x add complications and is normally not necessary. Using the same conditioning for all x, even when not optimal, is usually sufficient.

Example 4.2.2. *The following tiny example provides further insight into the choice of $\mathcal{G}$. Suppose X is the sum of two independent uniform random variables: $X = Y_1 + Y_2$ where $Y_1 \sim \mathcal{U}(0,1)$ and $Y_2 \sim \mathcal{U}(0,\epsilon)$ where $0 < \epsilon < 1$. The exact density of X here is $f(x) = x/\epsilon$ for $0 \le x \le \epsilon$, $f(x) = 1$ for $\epsilon \le x \le 1$, and $f(x) = (1+\epsilon-x)/\epsilon$ for $1 \le x \le 1+\epsilon$. Figure 4.1 illustrates this density.*

With $\mathcal{G} = \mathcal{G}_{-1}$, we have $F(x \mid \mathcal{G}_{-1}) = \mathbb{P}[X \le x \mid Y_2] = \mathbb{P}[Y_1 \le x - Y_2 \mid Y_2] = x - Y_2$ and the density estimator is $f(x \mid \mathcal{G}_{-1}) = 1$ for $Y_2 \le x \le 1 + Y_2$, and 0 elsewhere. If $\mathcal{G} = \mathcal{G}_{-2}$ instead, then $F(x \mid \mathcal{G}_{-2}) = \mathbb{P}[Y_2 \le x - Y_1 \mid Y_1] = (x - Y_1)/\epsilon$ and the density estimator is $f(x \mid \mathcal{G}_{-2}) = 1/\epsilon$ for $Y_1 \le x \le \epsilon + Y_1$, and 0 elsewhere. In both cases, the density estimator with one sample is a uniform density, but the second one is over a narrow interval if ϵ is small. When ϵ is small, $\mathcal{G} = \mathcal{G}_{-2}$ gives a density estimator $\hat{f}_{\text{cde},n}$ which is a sum of high narrow peaks and has much larger variance. For this simple example, we can also derive exact

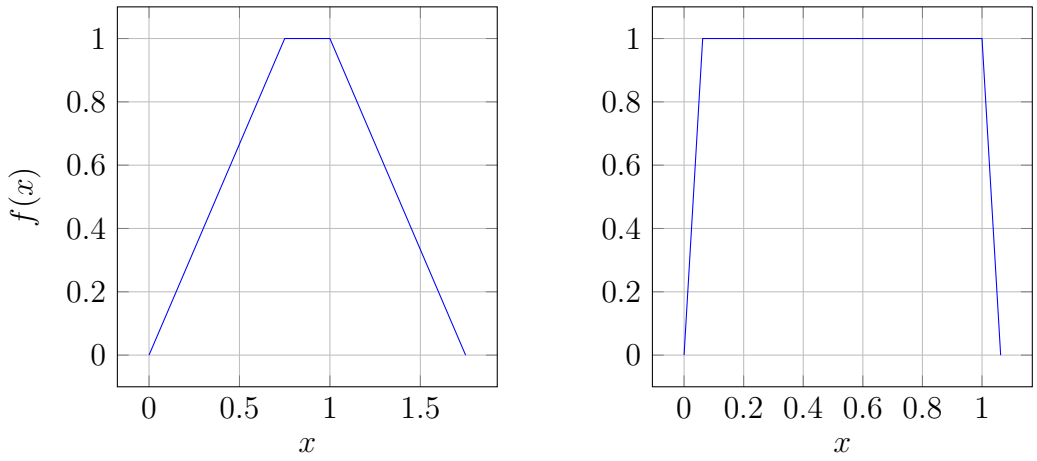

Figure 4.1. Exact density of X for the model in Example 4.2.2 with $\epsilon = 3/4$ (left) and $\epsilon = 1/16$ (right).

formulas for the IV of the CDE under MC. For $\mathcal{G} = \mathcal{G}_{-1}$, $f(x \mid \mathcal{G}_{-1}) = \mathbb{I}[Y_2 \le x \le 1 + Y_2]$ *is a Bernoulli random variable with mean* $\mathbb{P}[x - 1 \le Y_2 \le x] = f(x)$, *so its variance is* $f(x)(1 - f(x))$. *Integrating this over* $[0, 1 + \epsilon]$ *gives* $\mathrm{IV} = \epsilon/3$ *for one sample. For a sample of size* n, *this gives* $\mathrm{IV} = \epsilon/(3n)$. *For* $\mathcal{G} = \mathcal{G}_{-2}$, $f(x \mid \mathcal{G}_{-2}) = \mathbb{I}[Y_1 \le x \le \epsilon + Y_1]/\epsilon$ *has also mean* $f(x)$, *but its variance is* $\epsilon^{-1} f(x)(1 - \epsilon f(x))$ *which is larger than* $f(x)(1 - f(x))$ *when* ϵ *is small. Integrating over* $[0, 1 + \epsilon]$ *gives* $\mathrm{IV} = 1/\epsilon - 1 + \epsilon/3$ *for one sample, which is much larger than* $\epsilon/3$ *when* ϵ *is small. The take-away here: It is usually better to condition on lower-variance information and hide variables having a large variance contribution.*

Example 4.2.3. *In this example, we illustrate how Assumption 4.2.1 can be verified. Let* X *be the sum of two independent normal random variables,* $X = Y_1 + Y_2$, *where* $Y_1 \sim \mathcal{N}(0, \sigma_1^2)$, $Y_2 \sim \mathcal{N}(0, \sigma_2^2)$, *and* $\sigma_1^2 + \sigma_2^2 = 1$, *so* $X \sim \mathcal{N}(0, 1)$. *With* $\mathcal{G} = \mathcal{G}_{-2}$, *we have* $F(x \mid \mathcal{G}_{-2}) = \mathbb{P}[Y_2 \le x - Y_1] = \Phi((x - Y_1)/\sigma_2)$ *and the CDE is* $f(x \mid \mathcal{G}_{-2}) = \phi((x - Y_1)/\sigma_2)/\sigma_2$. *Assumption 4.2.1 is easily verified with* $\Gamma = \phi(0)/\sigma_2$ *and* $K_\gamma = \Gamma^2$, *so this estimator is unbiased for* $f(x) = \phi(x)$. *Its variance is*

$$\begin{aligned}
\mathrm{Var}[\phi((x - Y_1)/\sigma_2)/\sigma_2] &= \mathbb{E}[\exp[-(x - Y_1)^2/\sigma_2^2]/(2\pi\sigma_2^2)] - \phi^2(x) \\
&= \frac{1}{\sigma_2^2\sqrt{2\pi}} \mathbb{E}[\phi(\sqrt{2}(x - Y_1)/\sigma_2)] - \phi^2(x) \\
&= \frac{1}{\sigma_2\sqrt{2\pi(1 + \sigma_1^2)}} \phi\left(\sqrt{2}x/\sqrt{1 + \sigma_1^2}\right) - \phi^2(x). \qquad (4.2.6)
\end{aligned}$$

Example 4.2.4. *If X is the min or max of two or more continuous random variables, then in general $F(\cdot \mid \mathcal{G}_{-k})$ is not continuous, so Assumption 4.2.1 does not hold. To illustrate this, let $X = \max(Y_1, Y_2)$ where Y_1 and Y_2 are independent. With $\mathcal{G} = \mathcal{G}_{-2}$ (we hide Y_2), we have*

$$\mathbb{P}[X \le x \mid Y_1 = y] = \begin{cases} \mathbb{P}[Y_2 \le x \mid Y_1 = y] = F_2(x) & \text{if } x \ge y; \\ 0 & \text{if } x < y. \end{cases}$$

If $F_2(y) > 0$, this function is discontinuous at $x = y$. The same holds for the maximum of more than two variables. One way to handle this is to generate all the variables, then hide the maximum and compute its conditional density given the other ones. Without loss of generality, suppose Y_1 is the maximum and $Y_2 = x_2$ the second largest. Then the CDE of the max is $f(x|\mathcal{G}) = f_1(x|Y_1 > x_2)$ Note that for independent random variables whose cdfs and densities have an analytical form, the cdf and density of the max can often be computed analytically. See Section 4.4.5 for more on this. A very similar story holds if we replace the max by the min.

Example 4.2.5. *Suppose $X = Z.C$ where $Z \sim \mathcal{N}(0,1)$ and C is continuous with support over $(0,\infty)$. We can hide Z and generate $X \sim \mathcal{N}(0,C^2)$ conditional on C, or do the opposite. Which one is best depends on the distribution of C. Here we have $\mathrm{Var}[X] = \mathbb{E}[\mathrm{Var}[X|C]] = \mathbb{E}[C^2]$ while $\mathrm{Var}[\mathbb{E}[X|C]] = 0$. So the usual variance decomposition tells us nothing about what to hide. This illustrates the fact that there is rarely a simple rule to select the optimal $\mathcal{G}$.*

4.2.4. Convex combination of conditional density estimators

When there are many possible choices of $\mathcal{G}$ for a given problem, one may try to pick the best one, but sometimes a better approach is to select more than one and take a convex linear combination of the corresponding CDEs as the final density estimator. This idea is well known for general mean estimators (Bratley et al., 1987). More specifically, suppose $\hat{f}_{0,n}, \ldots, \hat{f}_{q,n}$ are $q+1$ distinct unbiased density estimators. Typically, these estimators will be dependent and will be based on the same simulations. They could be all CDEs based on different choices of $\mathcal{G}$ (so they will not hide the same information), but there could be

non-CDEs as well. A convex combination can take the form

$$\hat{f}_n(x) = \beta_0 \hat{f}_{0,n}(x) + \cdots + \beta_q \hat{f}_{q,n}(x) = \hat{f}_{0,n}(x) - \sum_{\ell=1}^{q} \beta_\ell (\hat{f}_{0,n}(x) - \hat{f}_{\ell,n}(x)) \tag{4.2.7}$$

for all $x \in \mathbb{R}$, where $\beta_0 + \cdots + \beta_q = 1$. This combination is equivalent to choosing $\hat{f}_{0,n}(x)$ as the main estimator, and taking the q differences $\hat{f}_{0,n}(x) - \hat{f}_{\ell,n}(x)$ as control variables (Bratley et al., 1987). With this interpretation, the optimal coefficients β_ℓ can be estimated via standard control variate theory (Asmussen and Glynn, 2007) by trying to minimize the IV of $\hat{f}_n(x)$ w.r.t. the β_ℓ's. More precisely, if we denote $\mathrm{IV}_\ell = \mathrm{IV}(\hat{f}_{\ell,n}(x))$ and

$$\mathrm{IC}_{\ell,k} = \int_a^b \mathrm{Cov}[\hat{f}_{\ell,n}(x), \hat{f}_{k,n}(x)] \mathrm{d}x,$$

we obtain

$$\mathrm{IV} = \mathrm{IV}\left(\hat{f}_n(x)\right) = \sum_{\ell=0}^{q} \beta_\ell^2 \mathrm{IV}_\ell + 2 \sum_{0 \le \ell < k \le q} \beta_\ell \beta_k \mathrm{IC}_{\ell,k}.$$

Given the IV_ℓ's and $\mathrm{IC}_{\ell,k}$'s (or good estimates of them), this IV is a quadratic function of the β_ℓ's, which can be minimized exactly as in standard least-squares linear regression, to obtain estimates of the optimal coefficients β_j. Estimating the density and coefficients from the same data yields biased but consistent density estimators, and the bias is rarely a problem. We did this for some of the examples in Section 6.4.

A more refined approach is to allow the coefficients β_j to depend on x:

$$\hat{f}_n(x) = \beta_0(x) \hat{f}_{0,n}(x) + \cdots + \beta_q(x) \hat{f}_{q,n}(x) = \hat{f}_{0,n}(x) - \sum_{\ell=1}^{q} \beta_\ell(x) (\hat{f}_{0,n}(x) - \hat{f}_{\ell,n}(x)), \tag{4.2.8}$$

where $\beta_0(x) + \cdots + \beta_q(x) = 1$ for all $x \in \mathbb{R}$. We can estimate the optimal coefficients by standard control variate theory at selected values of x, then for each $\ell \ge 1$, we can fit a smoothing spline to these estimated values, by least squares. This provides estimated optimal coefficients that are smooth functions of x, which can be used to obtain a final CDE. This type of strategy was used in L'Ecuyer and Buist (2008) to estimate varying control variate coefficients in a different setting. The additional flexibility can provide much more variance reduction in some situations.

4.2.5. A GLR density estimator (GLRDE)

The *generalized likelihood ratio* (GLR) method, originally developed by Peng et al. (2018) to estimate the derivative of an expectation with respect to some model parameter, can be adapted to density estimation, as shown in Peng et al. (2020). We summarize briefly here how this methode stimates the density $f(x)$ in our general setting, so we can apply it in our examples and make numerical comparisons. The assumptions stated below differ slightly from those in Peng et al. (2020). In particular, here we do not have a parameter θ, the conditions on the estimator are required only in the area where $X \le x$, and we add a condition to ensure finite variance. As in Section 4.2.3, we assume here that $X = h(\mathbf{Y}) = h(Y_1, \ldots, Y_d)$ where $Y_1, \ldots, Y_d$ are independent continuous random variables, and Y_j has cdf F_j and density f_j. Let $P(x) = \{\mathbf{y} \in \mathbb{R}^d : h(\mathbf{y}) \le x\}$. For $j = 1, \ldots, d$, let $h_j(\mathbf{y}) := \partial h(\mathbf{y})/\partial y_j$, $h_{jj}(\mathbf{y}) := \partial^2 h(\mathbf{y})/\partial y_j^2$, and

$$\Psi_j(\mathbf{y}) = \frac{\partial \log f_j(y_j)/\partial y_j - h_{jj}(\mathbf{y})/h_j(\mathbf{y})}{h_j(\mathbf{y})}. \tag{4.2.9}$$

Assumption 4.2.2. *The Lebesgue measure of $h^{-1}((x-\epsilon, x+\epsilon))$ in $\mathbb{R}^d$ goes to 0 when $\epsilon \to 0$ (this means essentially that the density is bounded around x).*

Assumption 4.2.3. *The set $P(x)$ is measurable, the functions h_j, h_{jj}, and Ψ_j are well defined over it, and $\mathbb{E}[\mathbb{I}[X \le x] \cdot \Psi_j^2(\mathbf{Y})] < \infty$.*

Proposition 4.2.2. *Under Assumptions 4.2.2 and 4.2.3, the GLRDE $\mathbb{I}[X \le x] \cdot \Psi_j(\mathbf{Y})$ is an unbiased and finite-variance estimator of the density $f(x)$ at x.*

Proof. For the proof of Proposition 4.2.2 and additional details, See Peng et al. (2020). □

4.3. Combining RQMC with the CMC density estimator

We now discuss how RQMC can be used with the CDE, and under what conditions it can provide a convergence rate faster than $\mathcal{O}(n^{-1})$ for the IV of the resulting unbiased estimator. For this, we first recall some basic facts about QMC and RQMC. More detailed coverages can be found in Niederreiter (1992), Dick and Pillichshammer (2010), and L'Ecuyer (2009, 2018), for example.

For a function $g : [0,1)^s \to \mathbb{R}$, the integration error by the average over a point set $P_n = \{\mathbf{u}_1, \ldots, \mathbf{u}_n\} \subset [0,1]^s$ is defined by

$$E_n = \frac{1}{n}\sum_{i=1}^{n} g(\mathbf{u}_i) - \int_{[0,1]^s} g(\mathbf{u})\mathrm{d}\mathbf{u}. \tag{4.3.1}$$

Classical QMC theory bounds this error as follows. Let $\mathfrak{v} \subseteq \mathcal{S} := \{1, \ldots, s\}$ denote an arbitrary subset of coordinates. For any point $\mathbf{u} = (u_1, \ldots, u_s) \in [0,1]^s$, $\mathbf{u}_\mathfrak{v}$ denotes the projection of $\mathbf{u}$ on the coordinates in $\mathfrak{v}$ and $(\mathbf{u}_\mathfrak{v}, \mathbf{1})$ is the point $\mathbf{u}$ in which u_j is replaced by 1 for each $j \notin \mathfrak{v}$. Let $g_\mathfrak{v} := \partial^{|\mathfrak{v}|} g / \partial \mathbf{u}_\mathfrak{v}$ denote the partial derivative of g with respect to all the coordinates in $\mathfrak{v}$. When $g_\mathfrak{v}$ exists and is continuous for $\mathfrak{v} = \mathcal{S}$ (i.e., for all $\mathfrak{v} \subseteq \mathcal{S}$), the *Hardy-Krause (HK) variation* of g can be written as

$$V_{\mathrm{HK}}(g) = \sum_{\emptyset \neq \mathfrak{v} \subseteq \mathcal{S}} \int_{[0,1]^{|\mathfrak{v}|}} \left| g_\mathfrak{v}(\mathbf{u}_\mathfrak{v}, \mathbf{1}) \right| \mathrm{d}\mathbf{u}_\mathfrak{v}. \tag{4.3.2}$$

On the other hand, the *star-discrepancy* of P_n is

$$D^*(P_n) = \sup_{\mathbf{u} \in [0,1]^s} \left| \frac{|P_n \cap [\mathbf{0}, \mathbf{u})|}{n} - \mathrm{vol}[\mathbf{0}, \mathbf{u}) \right|$$

where $\mathrm{vol}[\mathbf{0}, \mathbf{u})$ is the volume of the box $[\mathbf{0}, \mathbf{u})$. The classical *Koksma-Hlawka (KH) inequality* bounds the absolute error by the product of these two quantities, one that involves only the function g and the other that involves only the point set P_n:

$$|E_n| \leq V_{\mathrm{HK}}(g) \cdot D^*(P_n). \tag{4.3.3}$$

There are explicit construction methods (e.g., digital nets, lattice rules, and polynomial lattice rules) of deterministic point sets P_n for which $D^*(P_n) = \mathcal{O}((\log n)^{s-1}/n) = \mathcal{O}(n^{-1+\epsilon})$ for all $\epsilon > 0$. This means that functions g for which $V_{\mathrm{HK}}(g) < \infty$ can be integrated by QMC with a worst-case error that satisfies $|E_n| = \mathcal{O}(n^{-1+\epsilon})$. There are also known methods to randomize these point sets P_n in a way that each randomized point $\mathbf{u}_i$ has the uniform distribution over $[0,1)^s$, so $\mathbb{E}[E_n] = 0$, and the $\mathcal{O}(n^{-1+\epsilon})$ discrepancy bound is preserved, which gives

$$\mathrm{Var}[E_n] = \mathbb{E}[E_n^2] = \mathcal{O}(n^{-2+\epsilon}). \tag{4.3.4}$$

The classical definitions of variation and discrepancy given above are in fact only one pair among an infinite collection of possibilities. There are other versions of (4.3.3), with

different definitions of the discrepancy and the variation, such that there are known point set constructions for which the discrepancy converges as $\mathcal{O}(n^{-\alpha+\epsilon})$ for $\alpha > 1$, but the conditions on g to have finite variation are more restrictive (more smoothness is required) (Dick and Pillichshammer, 2010).

From a practical viewpoint, getting a good estimate or an upper bound on the variation of g that can be useful to bound the RQMC variance is a notoriously difficult problem. Even just showing that the variation is finite is not always easy. However, finite variation is not a necessary condition. In many realistic applications in which variation is known to be infinite, RQMC can nevertheless reduce the variance by a large factor (He and Wang, 2015; L'Ecuyer, 2009; L'Ecuyer and Munger, 2012). The appropriate explanation for this depends on the application. In many cases, part of the explanation is that the integrand g can be written as a sum of orthogonal functions (as in an ANOVA decomposition) and a set of terms in that sum have a large variance contribution and are smooth low-dimensional functions for which RQMC is very effective (L'Ecuyer, 2009; L'Ecuyer and Lemieux, 2000; Lemieux, 2009). Making such a decomposition and finding the important terms is usually difficult for realistic problems, but to apply RQMC in practice, this is not needed. The usual approach in applications is to try it and compare the RQMC variance with the MC variance empirically. We will do that in Section 6.4. To estimate the RQMC variance, we usually replicate the RQMC scheme n_r times independently, using the same point set but with n_r independent randomizations, then we compute the empirical mean and variance of the n_r independent realizations of the RQMC density estimator $(1/n)\sum_{i=1}^{n} g(\mathbf{U}_i)$.

To combine the CDE with RQMC, we must be able to write $F(x \mid \mathcal{G}) = \tilde{g}(x,\mathbf{u})$ and $f(x \mid \mathcal{G}) = \tilde{g}'(x,\mathbf{u}) = \mathrm{d}\tilde{g}(x,\mathbf{u})/\mathrm{d}x$ for some function $\tilde{g} : [a,b] \times [0,1)^s$. The function $\tilde{g}'(x,\cdot)$ will act as g in (4.3.1). The combined CDE+RQMC estimator $\hat{f}_{\text{cde-rqmc},n}(x)$ will be defined by

$$\hat{f}_{\text{cde-rqmc},n}(x) = \frac{1}{n}\sum_{i=1}^{n} \tilde{g}'(x,\mathbf{U}_i), \tag{4.3.5}$$

which is the RQMC version of (4.2.2). To estimate the RQMC variance, we can perform n_r independent randomizations to obtain n_r independent realizations of $\hat{f}_{\text{cde},n}$ in (4.2.2) with MC and n_r independent realizations of $\hat{f}_{\text{cde-rqmc},n}$ in (4.3.5) with RQMC, and compute the empirical IV. By putting together the previous results, we obtain:

Proposition 4.3.1. *If* $\sup_{x\in[a,b]} V_{HK}(\tilde{g}'(x,.)) < \infty$, *then with RQMC points sets* P_n *with* $D^*(P_n) = \mathcal{O}((\log n)^{s-1}/n)$, *for any* $\epsilon > 0$, *we have* $\sup_{x\in[a,b]} \mathrm{Var}[\tilde{f}_{cde-rqmc,n}(x)] = \mathcal{O}(n^{-2+\epsilon})$, *so the MISE of the CDE+RQMC estimator converges as* $\mathcal{O}(n^{-2+\epsilon})$.

This is rarely done in practic but it is instructive to illustrate how the HK variation of $\tilde{g}'(x,\cdot)$ can be bounded in our CDE setting, so that Proposition 4.3.1 applies. For this, we need to show that the integral of the partial derivative of $\tilde{g}'(x,\mathbf{u})$ with respect to each subset of coordinates of $\mathbf{u}$ is finite. This will prove that the variance bound (4.3.4) holds for the CDE for these examples. To prove the bounded HK variation, we need to take the partial derivative of $\tilde{g}'(x,\mathbf{u})$ with respect to each subset of coordinates of $\mathbf{u}$ and show that the integral of each such partial derivative is finite. We show how this can be done for some of our earlier examples.

Example 4.3.1. Consider a sum of random variables as in Example 4.2.1, with $\mathcal{G} = \mathcal{G}_{-k}$ summarized by the single real number S_{-k}. We have $F(x \mid \mathcal{G}) = F_k(x - S_{-k})$ and $f(x \mid \mathcal{G}) = f_k(x - S_{-k})$. Without loss of generality, let $k = d$. Suppose that each Y_j is generated by inversion from $U_j \sim \mathcal{U}(0,1)$, so $Y_j = F_j^{-1}(U_j)$ and $S_{-d} = F_1^{-1}(U_1) + \cdots + F_s^{-1}(U_s)$ with $s = d - 1$. This gives $\tilde{g}(x,\mathbf{U}) = F_d(x - S_{-d}) = F_d(x - F_1^{-1}(U_1) - \cdots - F_s^{-1}(U_s))$ and $\tilde{g}'(x,\mathbf{U}) = f_d(x - S_{-d}) = f_d(x - F_1^{-1}(U_1) - \cdots - F_s^{-1}(U_s))$. The partial derivatives of this last function are

$$\tilde{g}'_{\mathfrak{v}}(x, \mathbf{U}_{\mathfrak{v}}, \mathbf{1}) = f_d^{(|\mathfrak{v}|)}(x - S_{-d}) \prod_{j\in\mathfrak{v}} \frac{\partial(F_j^{-1}(U_j))}{\partial U_j}.$$

So the functions F_j^{-1} must be differentiable over $(0,1)$ for $j = 1,\ldots,d-1$, the density f_d must be s times differentiable, and the integral of $|\tilde{g}'_{\mathfrak{v}}(x, \mathbf{u}_{\mathfrak{v}}, \mathbf{1})|$ with respect to $\mathbf{u}_{\mathfrak{v}}$ must be bounded uniformly in $x \in [a,b]$. Under these conditions, the HK variation is bounded uniformly in x over $[a,b]$.

For Example 4.2.2, with $\mathcal{G} = \mathcal{G}_{-2}$ and $Y_1 = U_1$, we have $\tilde{g}'(x,\mathbf{u}) = \tilde{g}'(x,U_1) = \mathbb{I}[U_1 \le x \le \epsilon + U_1]/\epsilon = \mathbb{I}[x - \epsilon \le U_1 \le x]/\epsilon$. This function is not continuous, but its HK variation (not given by (4.3.2) in this case) is $2/\epsilon < \infty$, because it is piecewise constant with only two jumps, each one of size $1/\epsilon$. Thus, the HK variation is unbounded when $\epsilon \to 0$, but it is finite for any fixed ϵ. The behavior with $\mathcal{G} = \mathcal{G}_{-1}$ is similar and the HK variation is 2 in that case, which is much better.

For Example 4.2.3, if $\mathcal{G} = \mathcal{G}_{-2}$, we have $Y_1 = \sigma_1\Phi^{-1}(U_1)$ where $U_1 \sim \mathcal{U}(0,1)$. Then, $F(x \mid \mathcal{G}_{-2}) = F_2(x - Y_1) = \Phi((x - Y_1)/\sigma_2)$ and $f(x \mid \mathcal{G}_{-2}) = \phi((x - \sigma_1\Phi^{-1}(U_1))/\sigma_2)/\sigma_2 = \tilde{g}'(x,U_1)$.

Taking the derivative with respect to u and noting that $\mathrm{d}\Phi^{-1}(u)/\mathrm{d}u = 1/(\phi(\Phi^{-1}(u)))$ yields

$$\tilde{g}'_{\mathfrak{v}}(x,u) = \frac{\phi'((x-\sigma_1\Phi^{-1}(u))/\sigma_2)\,\sigma_1}{\sigma_2^2\phi(\Phi^{-1}(u))}$$

for $\mathfrak{v} = \{1\} = \mathcal{S}$ (the only subset in this case). Integrating this with respect to u by making the change of variable $z = \Phi^{-1}(u)$ gives

$$\int_0^1 \tilde{g}'_{\mathfrak{v}}(x,u)\mathrm{d}u = \frac{\sigma_1}{\sigma_2^2}\int_{-\infty}^{\infty} |\phi'((x-\sigma_1 z)/\sigma_2)|\,\mathrm{d}z < \infty,$$

because $|\phi'(\cdot)|$ is bounded by $\phi(\cdot)$ multiplied by the absolute value of a polynomial of degree 1. So the HK variation is bounded uniformly in x.

For all these examples in which the HK is unbounded, RQMC may still reduce the IV, but there is no guarantee. The GLRDE in Proposition 4.2.2 is typically discontinuous because of the indicator function, and therefore its HK variation is usually infinite.

4.4. Examples and numerical experiments

We now examine larger examples, summarize the results of our numerical experiments with the CDE and CDE+RQMC, and make comparisons with KDE and GLR, with MC and RQMC.

4.4.1. Experimental setting

Since the CDE is unbiased, we measure its performance by the IV, which equals the MISE in this case. To approximate the IV estimator (4.2.3) for a given n, we first take a stratified sample $e_1,\ldots,e_{n_e}$ of n_e *evaluation points* at which the empirical variance will be computed. We sample e_j uniformly in $[a+(j-1)(b-a)/n_e, a+j(b-a)/n_e)$ for $j = 1,\ldots,n_e$. Then we use the unbiased IV estimator

$$\widehat{\mathrm{IV}} = \frac{(b-a)}{n_e}\sum_{j=1}^{n_e}\widehat{\mathrm{Var}}[\hat{f}_n(e_j)],$$

where $\widehat{\mathrm{Var}}[\hat{f}_n(e_j)]$ is the empirical variance of the CDE at e_j, obtained as follows. We repeat the following n_r times, independently: Generate n observations of X from the density f with the given method (MC or RQMC), and compute the CDE at each evaluation point e_j. We then compute $\widehat{\mathrm{Var}}[\hat{f}_n(e_j)]$ as the empirical variance of the n_r density estimates at e_j, for each j. In all our examples, we used $n_r = 100$ and $n_e = 128$.

To estimate the convergence rate of the IV as a function of n with the different methods, we fit a model of the form $\mathrm{IV} \approx Kn^{-\nu}$. For the CDE with independent points (no RQMC), this model holds exactly with $\nu = 1$. We hope to observe $\nu > 1$ with RQMC. The parameters K and ν are estimated by linear regression in log-log scale, i.e., by fitting the model $\log \mathrm{IV} \approx \log K - \nu \log n$ to data. Since n is always taken as a power of 2, we report the logarithms in base 2. We estimated the IV for $n = 2^{14}, \ldots, 2^{19}$ (6 values) to fit the regression model. We also report the observed $-\log_2 \mathrm{IV}$ for $n = 2^{19}$ and use $e19$ as a shorthand for this value in the tables. We use exactly the same procedure for the GLRDE. For the KDE, these values are for the MISE instead of the IV. In all cases, we used a normal kernel and a bandwidth h selected by the methodology described in Ben Abdellah et al. (2019a). For some examples, we tried CDEs based on different choices of $\mathcal{G}$ and a convex combination as in Section 4.2.4. We report results with the following types of point sets:

(1) independent points (MC);
(2) a randomly-shifted lattice rule (Lat+s);
(3) a randomly-shifted lattice rule with a baker's transformation (Lat+s+b);
(4) Sobol' points with a left random matrix scramble and random digital shift (Sobol'+LMS).

The short names in parentheses are used in the plots and tables. For the definitions and properties of these RQMC point sets, see L'Ecuyer (2009, 2018); L'Ecuyer and Lemieux (2000); Owen (2003). They are implemented in SSJ (L'Ecuyer, 2016), which we used for our experiments. The parameters of the lattice rules were found with the Lattice Builder software of L'Ecuyer and Munger (2016), using a fast-CBC construction method with the $\mathcal{P}_2$ criterion and order dependent weights $\gamma_{\mathfrak{u}} = \rho^{|\mathfrak{u}|}$, with ρ ranging from 0.05 to 0.8, depending on the example (a larger ρ was used when the dimension s was smaller). The baker's transformation sometimes improves the convergence rate by making the integrand periodic (Hickernell, 2002), but it usually also increase the variation of the integrand, so its impact on the variance can go either way.

4.4.2. A sum of normals

We start with a very simple example in which the density f is known beforehand, so there is no real need to estimate it, but this type of example is very convenient for testing

the performance of our density estimators. Let $Z_1, \ldots, Z_d$ be independent standard normal random variables, i.e., with mean 0 and variance 1, and define

$$X = (a_1 Z_1 + \cdots + a_d Z_d)/\sigma, \qquad \text{where} \qquad \sigma^2 = a_1^2 + \cdots + a_d^2.$$

Then X is also standard normal, with density $f(x) = \phi(x) \stackrel{\text{def}}{=} \exp(-x^2/2)/\sqrt{2\pi}$ and cdf $\mathbb{P}[X \le x] = \Phi(x)$ for $x \in \mathbb{R}$. The term $a_j Z_j$ in the sum has variance a_j^2. We pretend we do not know this and we estimate $f(x)$ over the interval $[-2,2]$, which contains slightly more than 95% of the density. We also tried larger intervals, such as $[-5,5]$, and IVs for the CDE were almost the same.

To construct the CDE, we define $\mathcal{G}_{-k}$ as in Example 4.2.1, for any $k = 1, \ldots, d$. That is, we hide Z_k and estimate the cdf by

$$F(x \mid \mathcal{G}_{-k}) = \mathbb{P}\left[a_k Z_k \le x\sigma - \sum_{j=1,\, j\neq k}^{d} a_j Z_j \,\middle|\, \mathcal{G}_{-k} \right] = \Phi\left(\frac{x\sigma}{a_k} - \frac{1}{a_k} \sum_{j=1,\, j\neq k}^{d} a_j Z_j \right).$$

The CDE becomes

$$f(x \mid \mathcal{G}_{-k}) = \phi\left(\frac{x\sigma}{a_k} - \frac{1}{a_k} \sum_{j=1,\, j\neq k}^{d} a_j Z_j \right) \frac{\sigma}{a_k} = \phi\left(\frac{x\sigma}{a_k} - \frac{1}{a_k} \sum_{j=1,\, j\neq k}^{d} a_j \Phi^{-1}(U_j) \right) \frac{\sigma}{a_k} \stackrel{\text{def}}{=} \tilde{g}'(x, \mathbf{U})$$

for $x \in \mathbb{R}$, where $\mathbf{U} = (U_1, \ldots, U_{k-1}, U_{k+1}, \ldots, U_d)$, $Z_j = \Phi^{-1}(U_j)$, and the U_j are independent $\mathcal{U}(0,1)$ random variables. Assumption 4.2.1 is easily verified, so this CDE is unbiased.

For CMC+MC (independent sampling), we get an exact formula for the variance of the CDE directly from Example 4.2.3, by taking in that example $Y_2 = a_k Z_k/\sigma$ and $Y_1 = X - Y_2$, whose variances are $\sigma_2^2 = (a_k/\sigma)^2$ and $\sigma_1^2 = 1 - \sigma_2^2$, and plugging these values into (4.2.6). To prove that RQMC gives a better convergence rate for the variance than MC, it suffices to show that $V_{\text{HK}}(\tilde{g}'(x,\cdot)) < \infty$ for any x. This can be done by the same argument as in the second part of Example 4.3.1. Then we expect to observe a convergence rate near $\mathcal{O}(n^{-2})$, at least when d is small.

For the GLRDE method, with $Y_j = Z_j a_j/\sigma \sim \mathcal{N}(0, a_j^2/\sigma^2)$, we obtain $\partial(\log f_j(y_j))/\partial y_j = -y_j\sigma^2/a_j^2$, $h_j(y_j) = 1$, $h_{jj}(y_j) = 0$, and then $\Psi_j = -Y_j\sigma^2/a_j^2 = -Z_j\sigma/a_j$. Note that we could also replace Y_j by Z_j and f_j by ϕ_j (the standard normal density), which would give $\partial(\log \phi_j(z_j))/\partial z_j = -z_j$, $h_j(z_j) = a_j/\sigma$, $h_{jj}(y_j) = 0$, and again $\Psi_j = -Z_j\sigma/a_j$.

In our first experiment, we take $a_j = 1$ for all j, and $k = d$. By symmetry, the true IV is the same for any other k. Table 4.1 reports the estimated rate $\hat{\nu}$ and the estimated value

		$\hat{\nu}$					e19				
		$d=2$	$d=3$	$d=5$	$d=10$	$d=20$	$d=2$	$d=3$	$d=5$	$d=10$	$d=20$
CDE-1	MC	0.99	0.98	1.02	1.00	1.02	22.1	21.4	20.8	19.8	19.2
	Lat+s	2.83	2.00	1.85	1.40	1.04	52.3	39.8	32.1	23.6	19.7
	Lat+s+b	2.69	2.11	1.69	1.14	1.05	50.5	41.5	31.1	21.8	20.0
	Sob+LMS	2.62	2.10	1.81	1.04	1.04	49.3	40.7	31.1	21.3	19.7
CDE-avg	MC	1.06	0.92	1.03	1.01	1.01	23.4	22.1	21.6	20.6	19.8
	Lat+s	2.79	1.84	1.33	1.19	1.05	53.3	39.8	32.2	23.0	20.6
	Lat+s+b	2.65	1.90	1.71	1.05	1.08	51.6	41.4	32.3	23.4	21.3
	Sob+LMS	2.60	2.10	1.92	1.02	1.03	49.8	42.0	33.0	22.7	20.5
GLRDE	MC	0.98	0.95	1.03	1.05	1.00	17.0	16.1	15.9	14.9	14.1
	Lat+s	1.51	1.56	1.45	0.94	1.06	28.2	24.9	22.1	17.8	17.2
	Lat+s+b	1.49	1.41	1.05	1.06	1.04	27.3	23.9	20.4	18.8	17.6
	Sob+LMS	1.49	1.33	1.15	0.99	1.16	27.5	24.0	21.0	18.3	17.4
KDE	MC	0.79	0.80	0.76	0.75	0.77	17.0	17.0	16.9	16.9	17.0
	Lat+s	1.08	1.39	0.92	0.97	0.76	25.1	22.4	19.4	18.2	17.4
	Lat+s+b	1.23	0.94	0.72	0.73	0.74	24.1	20.1	18.1	17.3	17.2
	Sob+LMS	1.18	0.98	0.83	0.74	0.77	24.4	20.8	17.9	17.2	17.1

Table 4.1. Values of $\hat{\nu}$ and e19 for a CDE, and a convex combination of CDEs, a GLRDE, and a KDE, for a sum of $d = k$ normals with $a_j = 1$, over $[-2,2]$.

of e19 $= -\log_2(\text{IV})$ for $n = 2^{19}$, for various values of d and sampling methods. The rows marked CDE-1 give the results for $k = d$, while those labeled CDE-Avg are for a convex combination (4.2.7) with equal weights $\beta_\ell = 1/d$ for all $\ell = k - 1$, after computing the CDE for each k from the same simulations.

For MC, the rates $\hat{\nu}$ agree with the (known) exact asymptotic rates of $\nu = 1$ for the CDE and GLR, and $\nu = 0.8$ for the KDE. By looking at e19, we see that the MISE with MC is much smaller for the CDE than for the GLRDE and KDE, for example for $d = 2$ by a factor of about 32 for CDE-1 and about 70 for CDE-avg. For $d = 20$, the gains are more modest. RQMC methods provide huge improvements for small d with the CDE. We observe rates $\hat{\nu}$ larger than 2 for $d = 2$ and 3, and by looking at the exponents e19, we see that for $d = 3$, for example, the MISE goes from 2^{-17} for the GLRDE and KDE to about 2^{-42} for CDE-1 with Sobol' points with LMS. This is a reduction factor of about $2^{25} \approx 33$ millions for $n = 2^{19}$. The large values of $\hat{\nu}$ imply of course that this factor is smaller for smaller n. When d is large, such as $d = 20$, RQMC brings only a small gain. The values of $\hat{\nu}$ are sometimes noisy. For GLRDE with Lat+s and $d = 5$, for example, the large $\hat{\nu} = 1.45$ comes from the fact that the IV for $n = 2^{14}$ (not shown) is unusually large (an outlier). Looking at e19 gives a more robust assessment of the performance. GLRDE performs better than the KDE under

RQMC for small d, but not comparable to the CDE. Under MC, GLRDE is slightly worse than the KDE.

	$\hat{\nu}$				e19			
	$k=1$	$k=2$	$k=5$	$k=11$	$k=1$	$k=2$	$k=5$	$k=11$
MC	1.00	1.02	1.01	1.00	22.2	21.0	18.8	15.5
Lat+s	1.43	1.48	1.34	1.04	30.3	28.5	22.8	15.6
Lat+s+b	1.57	1.65	1.28	1.02	33.5	30.8	22.1	15.6
Sob+LMS	1.78	1.56	1.21	1.02	34.1	30.4	21.7	15.7

Table 4.2. Values of $\hat{\nu}$ and e19 with a CDE for selected choices of $\mathcal{G}_{-k}$, for a linear combination of $d = 11$ normals with $a_j^2 = 2^{1-j}$.

In our second experiment, we take $a_j^2 = 2^{1-j}$ for $j = 1, \ldots, d$. Now, the choice of k for the CDE makes a difference, and the best choice will obviously be $k = 1$, i.e., hide the term that has the largest variance. Note that with MC, $\mathrm{Var}[X] = 2 - 2^{-d}$, and when we apply CMC by hiding $a_k Z_k$ from the sum, we hide a term of variance $a_k^2 = 2^{1-k}$ and generate a partial sum S_{-k} of variance $2 - 2^{1-k} - 2^{-d}$. Both terms have a normal distribution with mean 0. The results of Example 4.2.3 hold with these variances. Table 4.2 reports the numerical results for $d = 11$ and $k = 1, 2, 5, 11$.

The MC rates $\hat{\nu}$ agree again with the theory, but here the IV depends very much on the choice of k, and this effect is more significant when k is smaller. For example, for Sobol' points, the IV with $k = 1$ is about 300,000 times smaller than with $k = 11$. The reason is that with $k = 11$, we hide only a variable having a very small variance, so the CDE for one sample is a high narrow peak, and the HK variation of $\tilde{g}'(x,\mathbf{u})$ is very large. For $k = 1$ or 2, we have the opposite and the integrand is much more RQMC-friendly.

4.4.3. Displacement of a cantilever beam

We consider the following model for the displacement X of a cantilever beam with horizontal and vertical loads, taken from Bingham (2017):

$$X = h(Y_1,Y_2,Y_3) = \frac{4\ell^3}{Y_1 w t}\sqrt{\frac{Y_2^2}{w^4} + \frac{Y_3^2}{t^4}} \tag{4.4.1}$$

in which $\ell = 100$, $w = 4$ and $t = 2$ are constants (in inches), while Y_1 (Young's modulus), Y_2 (the horizontal load), and Y_3 (the vertical load), are independent normal random variables, $Y_j \sim \mathcal{N}(\mu_j,\sigma_j^2)$, i.e., normal with mean μ_j and variance σ_j^2. The parameter values are

$\mu_1 = 2.9 \times 10^7$, $\sigma_1 = 1.45 \times 10^6$, $\mu_2 = 500$, $\sigma_2 = 100$, $\mu_3 = 1000$, $\sigma_3 = 100$. We will denote $\kappa = 4\ell^3/(wt) = 5 \times 10^5$. The goal is to estimate the density of X over the interval $[3.1707, 5.6675]$, which covers about 99% of the density (it clips 0.5% on each side). It is possible to have $X < 0$ in this model, but the probability is $\mathbb{P}[Y_1 < 0] = \Phi(-20) = 2.8 \times 10^{-89}$, which is negligible. This example fits the framework of Section 4.2.3, with $d = 3$. We can hide any of the three random variables for the conditioning, and we will examine each case.

Conditioning on $\mathcal{G}_{-1}$ means hiding Y_1. We have

$$X = \frac{\kappa}{Y_1}\sqrt{\frac{Y_2^2}{w^4} + \frac{Y_3^2}{t^4}} \le x \quad \text{if and only if} \quad Y_1 \ge \frac{\kappa}{x}\sqrt{\frac{Y_2^2}{w^4} + \frac{Y_3^2}{t^4}} \stackrel{\text{def}}{=} W_1(x).$$

Note that $W_1(x) > 0$ if and only if $x > 0$. For $x > 0$,

$$F(x \mid \mathcal{G}_{-1}) = \mathbb{P}[Y_1 \ge W_1(x) \mid W_1(x)] = 1 - \Phi((W_1(x) - \mu_1)/\sigma_1)$$

which is continuous and differentiable in x, and

$$f(x \mid \mathcal{G}_{-1}) = -\phi((W_1(x) - \mu_1)/\sigma_1)W_1'(x)/\sigma_1 = \phi((W_1(x) - \mu_1)/\sigma_1)W_1(x)/(x\sigma_1).$$

If we condition on $\mathcal{G}_{-2}$ instead, i.e., we hide Y_2, we have $X \le x$ if and only if

$$Y_2^2 \le w^4\left((xY_1/\kappa)^2 - Y_3^2/t^4\right) \stackrel{\text{def}}{=} W_2(x).$$

If $W_2(x) \le 0$, then $F'(x \mid \mathcal{G}_{-2}) = F(x \mid \mathcal{G}_{-2}) = \mathbb{P}[X \le x \mid W_2(x)] = 0$. For $W_2(x) > 0$, we have

$$\begin{aligned} F(x \mid \mathcal{G}_{-2}) &= \mathbb{P}[X \le x \mid W_2(x)] = \mathbb{P}\left[-\sqrt{W_2(x)} \le Y_2 \le \sqrt{W_2(x)} \mid W_2(x)\right] \\ &= \Phi((\sqrt{W_2(x)} - \mu_2)/\sigma_2) - \Phi(-(\sqrt{W_2(x)} + \mu_2)/\sigma_2), \end{aligned}$$

which is again continuous and differentiable in x, and

$$f(x \mid \mathcal{G}_{-2}) = \frac{\phi((\sqrt{W_2(x)} - \mu_2)/\sigma_2) + \phi(-(\sqrt{W_2(x)} + \mu_2)/\sigma_2)}{(\sigma_2\sqrt{W_2(x)})/(w^4 x(Y_1/\kappa)^2)} > 0.$$

If we condition on $\mathcal{G}_{-3}$, the analysis is the same as for $\mathcal{G}_{-2}$, by symmetry, and we get

$$f(x \mid \mathcal{G}_{-3}) = \frac{\phi((\sqrt{W_3(x)} - \mu_3)/\sigma_3) + \phi(-(\sqrt{W_3(x)} + \mu_3)/\sigma_3)}{(\sigma_3\sqrt{W_3(x)})/(t^4 x(Y_1/\kappa)^2)} > 0$$

for $W_3(x) > 0$, where $W_3(x)$ is defined in a similar way as $W_2(x)$. In addition to testing these three ways of conditioning, we also tested a convex combination of the three, as explained in Section 4.2.4, with coefficients β_ℓ that do not depend on x.

For GLRDE using Y_1, let $C = C(Y_2, Y_3) = (4\ell^3/wt)\sqrt{Y_2^2/w^4 + Y_3^2/t^4}$. Then, we have $X = h(\mathbf{Y}) = C/Y_1$, $h_1(\mathbf{Y}) = -CY_1^{-2}$, $h_{11}(\mathbf{Y}) = 2CY_1^{-3}$, $\partial \log f_1(Y_1)/\partial Y_1 = (Y_1 - \mu_1)/\sigma_1^2$, and

$$\Psi_1 = \frac{Y_1}{C}\left(Y_1(Y_1 - \mu_1)/\sigma_1^2 - 2\right).$$

	$\hat{\nu}$						e19					
	$\mathcal{G}_{-1}$	$\mathcal{G}_{-2}$	$\mathcal{G}_{-3}$	comb.	GLRDE	KDE	$\mathcal{G}_{-1}$	$\mathcal{G}_{-2}$	$\mathcal{G}_{-3}$	comb.	GLRDE	KDE
MC	0.97	0.98	0.99	0.98	1.02	0.76	19.3	14.5	22.8	22.5	14.1	15.8
Lat+s	1.99	1.95	2.06	2.04	1.38	1.03	39.8	25.2	41.6	41.9	23.4	21.9
Lat+s+b	2.24	2.08	2.27	2.25	1.37	0.93	44.5	23.7	46.8	47.0	23.3	21.0
Sob+LMS	2.21	2.03	2.21	2.21	1.32	0.97	44.0	23.6	45.7	46.1	23.4	21.5

Table 4.3. Values of $\hat{\nu}$ and e19 with a CDE for each choice of $\mathcal{G}_{-k}$, for the best convex combination, for the GLRDE, and for the KDE, for the cantilever beam model.

Table 4.3 summarizes the results. The MISE is about 2^{-47} for the best CDE+RQMC compared with $2^{-15.8}$ for usual KDE+MC, a gain by a factor of over $2^{31} \approx 2$ billions. With RQMC, the convergence rate $\hat{\nu}$ is around 2 in all cases with the CDE methods, and much less for GLRDE and KDE. GLRDE benefits significantly from RQMC, more than the KDE, but cannot compete with the CDE. For the lattice rules, the baker's transformation helps significantly for the CDE.

We also see that conditioning on $\mathcal{G}_{-2}$ does not give as much reduction than for the other choices. The reason is that the conditional density in this case is a high narrow peak, similar to what we saw for $k = 11$ at the end of Example 4.4.2. To provide visual insight, Figure 4.2 shows plots of five realizations of the conditional density for $\mathcal{G}_{-1}$, $\mathcal{G}_{-2}$, and $\mathcal{G}_{-3}$. The realizations of $f(\cdot \mid \mathcal{G}_{-2})$ have high narrow peaks. The average of the five realizations is shown in red and the true density in black. In Figure 4.3, we zoom in on part of the estimated densities to show the difference between MC and RQMC. In each panel one can see the CDE using MC (in red), using RQMC (in green), and the "true density" (black, dashed) estimated with RQMC using a very large number of samples. We have $\mathcal{G}_{-1}$ with $n = 2^{10}$ on the left and $\mathcal{G}_{-2}$ with $n = 2^{16}$ on the right. In both cases, the RQMC estimate

is closer to the true density, and on the right it oscillates less. If we repeat this experiment several times, the red curve would vary much more than the green one across the realizations.

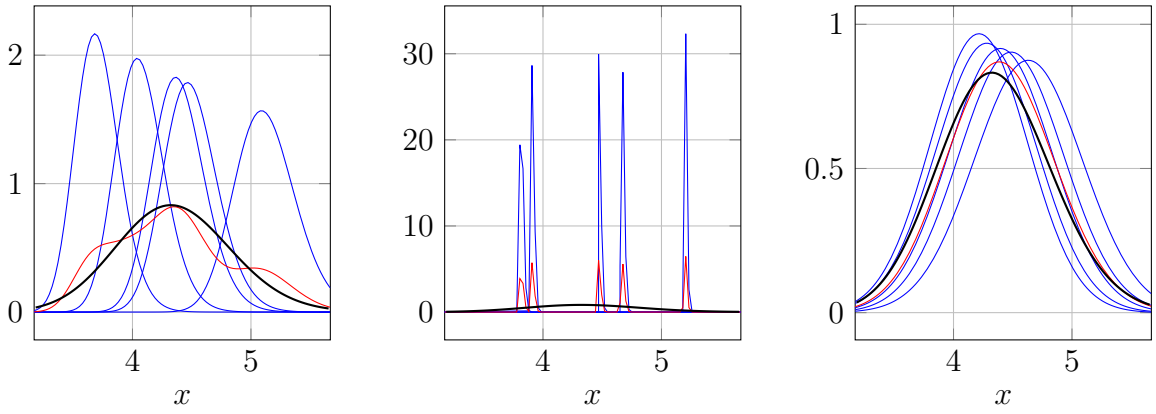

Figure 4.2. Five realizations of the density conditional on $\mathcal{G}_{-k}$ (blue), their average (red), and the true density (thick black) for $k = 1$ (left), $k = 2$ (middle), and $k = 3$ (right), for the cantilever example.

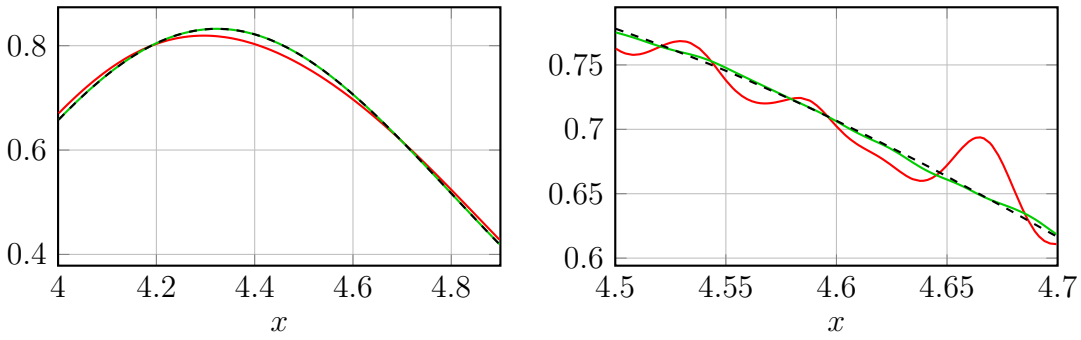

Figure 4.3. The CDE under MC (red), under RQMC (green) and the true density (black, dashed) for $\mathcal{G}_{-1}$ with $n = 2^{10}$ (left) and for $\mathcal{G}_{-2}$ with $n = 2^{16}$ (right), for the cantilever example.

4.4.4. Buckling strength of a steel plate

This is a six-dimensional example, taken from Schields and Zhang (2016). It models the buckling strength of a steel plate by

$$X = \left(\frac{2.1}{\Lambda} - \frac{0.9}{\Lambda^2}\right)\left(1 - \frac{0.75Y_5}{\Lambda}\right)\left(1 - \frac{2Y_6Y_2}{Y_1}\right), \tag{4.4.2}$$

where $\Lambda = (Y_1/Y_2)\sqrt{Y_3/Y_4}$, and $Y_1, \ldots, Y_6$ are independent random variables whose distributions are given in Table 4.4. Each distribution is either normal or lognormal, and the table gives the mean and the coefficient of variation (cv), which is the standard deviation divided

parameter	distribution	mean	cv
Y_1	normal	23.808	0.028
Y_2	lognormal	0.525	0.044
Y_3	lognormal	44.2	0.1235
Y_4	normal	28623	0.076
Y_5	normal	0.35	0.05
Y_6	normal	5.25	0.07

Table 4.4. Distribution of each parameter for the buckling strength model.

by the mean. We estimate the density of X over $[a,b] = [0.5169,0.6511]$, which contains about 99% of the density (leaving out 0.5% on each side). There is a nonzero probability of having $Y_4 \le 0$, in which case X is undefined, but this probability is extremely small and this has a negligible impact on the density estimator over $[a,b]$, so we just ignore it (alternatively we could truncate the density of Y_4). There are also negligible probabilities that the density estimates below are negative and we ignore this.

For this example, computing the density of X conditional on $\mathcal{G}_{-5}$ or $\mathcal{G}_{-6}$ (i.e., when hiding Y_5 or Y_6) is relatively easy, so we will try and compare these two choices. If we hide one of the variables that appear in Λ, the CDE would be harder to compute (it would require to solve a polynomial equation of degree 4 for each sample), and we do not do it. Let us define

$$V_1 = \frac{2.1}{\Lambda} - \frac{0.9}{\Lambda^2}, \qquad V_2 = 1 - \frac{2Y_6Y_2}{Y_1}, \quad \text{and} \quad V_3 = 1 - \frac{3Y_5}{4\Lambda}.$$

Then we have

$$X \le x \quad \Leftrightarrow \quad Y_5 \ge \left(1 - \frac{x}{V_1V_2}\right)\frac{4\Lambda}{3}$$

and

$$f(x \mid \mathcal{G}_{-5}) = f_5\left(\left(1 - \frac{x}{V_1V_2}\right)\frac{4\Lambda}{3}\right)\frac{4\Lambda}{3V_1V_2} = \phi\left(\frac{(1 - x/(V_1V_2))\,4\Lambda/3 - 0.35}{0.0175}\right)\frac{4\Lambda}{0.0525 \cdot V_1V_2}.$$

Similarly,

$$f(x \mid \mathcal{G}_{-6}) = f_6\left(\left(1 - \frac{x}{V_1V_3}\right)\frac{Y_1}{2Y_2}\right)\frac{Y_1}{2Y_2V_1V_3} = \phi\left(\frac{(1 - x/(V_1V_3))\,Y_1/(2Y_2) - 5.25}{0.3675}\right)\frac{Y_1}{0.735 \cdot Y_2V_1V_3}.$$

For GLRDE using Y_6, let $C = (2.1/\Lambda - 0.9/\Lambda^2)\,(1 - 0.75Y_5/\Lambda)$. We have $X = h(\mathbf{Y}) = C(1 - 2Y_6Y_2/Y_1)$, $h_6(\mathbf{Y}) = 2CY_2/Y_1$, $h_{66}(\mathbf{Y}) = 0$, $\partial \log f_6(Y_6)/\partial Y_6 = -(Y_6 - \mu_6)/\sigma_6^2$, and $\Psi_6 = Y_1(Y_6 - \mu_6)/(2CY_2\sigma_6^2)$.

	$\hat{\nu}$					e19				
	$\mathcal{G}_{-5}$	$\mathcal{G}_{-6}$	comb.	GLRDE	KDE	$\mathcal{G}_{-5}$	$\mathcal{G}_{-6}$	comb.	GLRDE	KDE
MC	1.00	1.00	1.00	0.98	0.76	13.5	15.4	15.4	10.2	11.7
Lat+s	1.89	1.56	1.56	1.29	0.81	20.0	24.9	24.9	16.6	13.7
Lat+s+b	1.46	1.65	1.60	1.19	0.85	17.5	25.1	25.1	15.9	12.7
Sob+LMS	1.40	1.75	1.75	1.16	0.81	17.7	25.5	25.5	15.9	12.4

Table 4.5. Values of $\hat{\nu}$ and e19 with a CDE for $\mathcal{G}_{-5}$, $\mathcal{G}_{-6}$, their combination, GLRDE, and the KDE, for the buckling strength model.

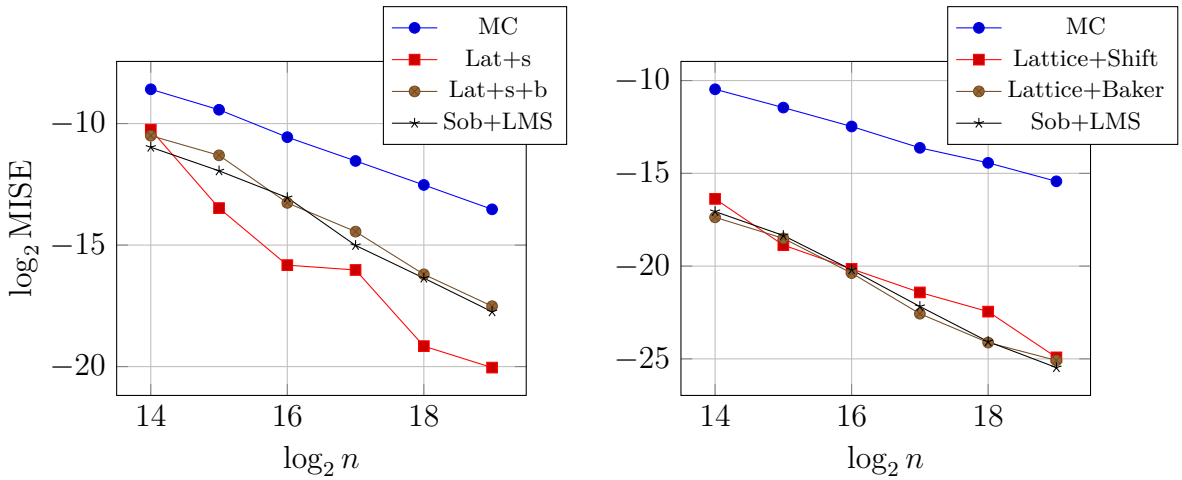

Figure 4.4. MISE vs n in log-log scale for the $\mathcal{G} = \mathcal{G}_{-5}$ (left) and $\mathcal{G} = \mathcal{G}_{-6}$ (right) for the buckling strength model.

Table 4.5 summarizes the results. We see again that the CDE with RQMC performs very well and much better than the GLRDE and KDE, that it is much better to condition on $\mathcal{G}_{-6}$ than on $\mathcal{G}_{-5}$, and that combining the two provides no significant improvement. The GLRDE is also better than the KDE under RQMC, but not under MC. Figure 4.4 displays the IV as a function of n in a log-log-scale for the CDE with $\mathcal{G}_{-5}$ and $\mathcal{G}_{-6}$. It unveils a slightly more erratic behavior of the MISE for the shifted lattice rule (Lat+s) than for the other methods; the performance depends on the choice of parameters of the lattice rule and their interaction with the particular integrand.

4.4.5. A stochastic activity network

In this example, the conditioning must hide more than one random variable. We consider an acyclic directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ where $\mathcal{N}$ is a finite set of nodes and $\mathcal{A} = \{a_j = (\alpha_j, \beta_j),\ j = 1, \ldots, d\}$ a finite set of arcs (directed links) where a_j goes from α_j to β_j. There is a source node having only outcoming arcs and a sink node having only incoming links, and

each arc belongs to at least one path going from the source to the sink. There can be at most one arc for each pair (α_j, β_j) (no parallel arcs). Each arc j has random length Y_j. These Y_j are assumed independent with continuous cdf's F_j, density f_j, and can be generated by inversion: $Y_j = F_j^{-1}(U_j)$ where $U_j \sim U(0,1)$. The length of the longest path from the source to the sink is a random variable X and the goal is to estimate the density of X.

This general model has several applications. The arcs a_j may represent activities having random durations and the graph represents precedence relationships between all activities of a project. Activity a_j cannot start before all activities j' with $\beta_{j'} = \alpha_j$ are completed. Then X represents the duration of the project if all activities are started as soon as allowed. This type of *stochastic activity network* (SAN) is widely used in project management for all types of projects (e.g., construction, software, etc.), communication, transportation, etc. For example, the graph may represent a large railway network in which each activity corresponds to a train stopping at a station, or a train covering a given segment of its route, or a minimal spacing between trains, etc. Precedence relationships are needed because railways are shared, there are ordering and distancing rules between trains, passengers have connections between trains, trains are merged or split at certain points, etc. The travel time of one passenger in this network turns out to be the length X of the longest path in a subnetwork whose source and sink are the origin and destination of this passenger.

For our numerical experiments, we use a small example from Avramidis and Wilson (1996, 1998), who showed how to use CMC to estimate $\mathbb{E}[X]$ and some quantiles of the distribution of X. L'Ecuyer and Lemieux (2000) and L'Ecuyer and Munger (2012) used this same example to test the combination of CMC with RQMC to estimate $\mathbb{E}[X]$. The network is depicted in Figure 4.5 and the cdf's F_j are given in Avramidis and Wilson (1996) and L'Ecuyer and Munger (2012). We will estimate the density of X over $[a,b] = [22,106.24]$, which covers about 95% of the density.

Here, X is defined as the maximum length over several paths, and if we hide only a single random variable Y_j to implement the CDE, we run into the same problem as in Example 4.2.4: Assumption 4.2.1 does not hold, because $F(.|\mathcal{G})$ has a jump. This means that we must hide more information (condition on less). Following Avramidis and Wilson (1996, 1998), we select a uniformly directed cut $\mathcal{L}$, which is a set of activities such that each path from the source to the sink contains exactly one activity from $\mathcal{L}$, and let $\mathcal{G}$ represent $\{Y_j, j \notin \mathcal{L}\}$. In

Figure 4.5, {1,2}, {11,13}, {5,6,7,9,10}, and {2,3,5,8,9,13}, are all valid choices of $\mathcal{L}$. The corresponding conditional cdf is

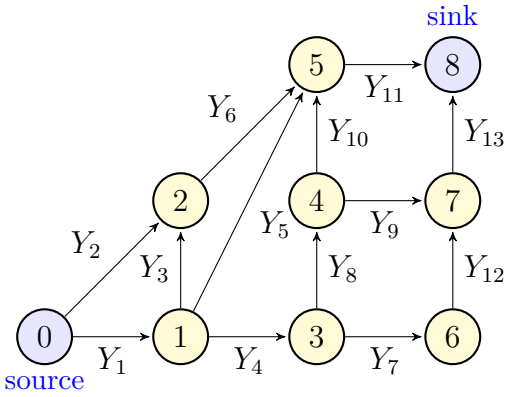

Figure 4.5. A stochastic activity network

$$F(x \mid \mathcal{G}) = \mathbb{P}\left[X < x \mid \{Y_j : j \notin \mathcal{L}\}\right] = \prod_{j \in \mathcal{L}} \mathbb{P}[Y_j \le x - P_j] = \prod_{j \in \mathcal{L}} F_j(x - P_j) \tag{4.4.3}$$

where P_j is the length of the longest path that goes through arc j when we exclude Y_j from that length. The conditional density is

$$f(x \mid \mathcal{G}) = \frac{\mathrm{d}}{\mathrm{d}x} F(x \mid \mathcal{G}) = \sum_{j \in \mathcal{L}} f_j(x - P_j) \prod_{l \in \mathcal{L}, l \neq j} F_l(x - P_j).$$

Under this conditioning, since the Y_j's are continuous variables with bounded variance, Assumption 4.2.1 holds, so $f(x \mid \mathcal{G})$ is an unbiased density estimator with uniformly bounded variance.

For our numerical experiments, we use the same cut $\mathcal{L} = \{5,6,7,9,10\}$ as Avramidis and Wilson (1996), indicated in light blue in Figure 4.5, even though there are other cuts with six links, which could possibly perform better because they hide more links. We could also compute the CDE with several choices of $\mathcal{L}$ and then take a convex combination.

The GLRDE method described in Section 4.2.5 does not work for this example. Indeed, with $X = h(\mathbf{Y})$ defined as the length of the longest path, for any j, the derivative $h_j(\mathbf{Y})$ is zero whenever arc j is not on the longest path, so we would need to select an arc j that is guaranteed to be always on the longest path. But there is no such arc. We could apply a modified GLRDE that selects a cut instead of a single coordinate Y_j, but this is beyond the scope of this paper.

		$\hat{\nu}$	e19
CDE	MC	0.96	25.6
	Lat+s	1.31	30.9
	Lat+s+b	1.17	29.6
	Sob+LMS	1.27	29.9
KDE	MC	0.78	20.9
	Lat+s	0.95	22.7
	Lat+s+b	0.93	22.0
	Sob+LMS	0.74	21.9

Table 4.6. Values of $\hat{\nu}$ and e19 with the CDE and KDE, for the SAN example.

Table 4.6 and Figure 4.6 summarize our results. We see that for $n = 2^{19}$, the CDE outperforms the KDE by a factor of about 20 with MC, and by a factor of about $2^8 \approx 250$ with RQMC. Interestingly, here the lattice rules work better without the baker's transformation.

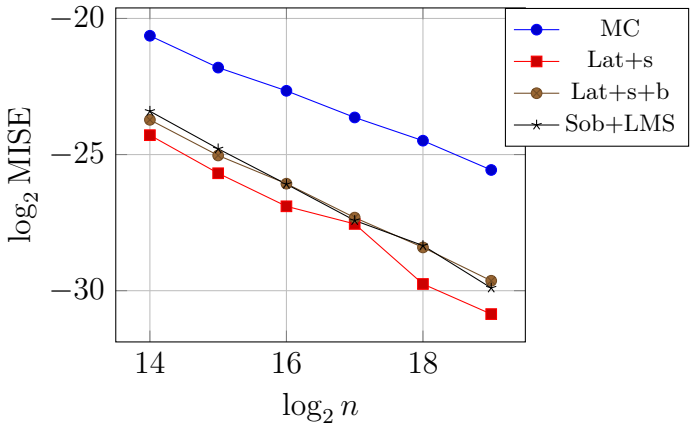

Figure 4.6. MISE vs n in log-log scale, for the SAN example.

4.4.6. Density of the failure time of a system

We consider a d-component system in which each component starts in the operating mode (state 1) and fails (jumps to state 0) at a certain random time, to stay there forever. Let Y_j be the failure time of component j for $j = 1, \ldots, d$. For $t \geq 0$, let $W_j(t) = \mathbb{I}[Y_j > t]$ be the state of component j and $\mathbf{W}(t) = (W_1(t), \ldots, W_d(t))^\mathsf{t}$ the system state, at time t. The system is in the failed mode at time t if and only if $\Phi(\mathbf{W}(t)) = 0$, where $\Phi : \{0,1\}^d \to \{0,1\}$ is called the *structure function*. Let $X = \inf\{t \geq 0 : \Phi(\mathbf{W}(t)) = 0\}$ be the random time when the system fails. We want to estimate the density of X. A straightforward way of simulating a realization of X is to generate the component lifetimes $Y_j = \inf\{t \geq 0 : W_j(t) = 0\}$ for $j = 1, \ldots, d$, and then compute X from that.

As in Section 4.4.5, the GLRDE method of Section 4.2.5 does not work for this example, because $h_j(\mathbf{Y}) \neq 0$ only when $X = Y_j$, and there is no j for which this is sure to happen.

If the Y_j are independent and exponential, one can construct a CMC estimator of the cdf $F(x) = \mathbb{P}[X \leq x]$ as follows (Botev et al., 2013, 2016; Gertsbakh and Shpungin, 2010). Generate all the Y_j's and sort them in increasing order. Then, erase their values and retain only their order, which is a permutation π of $\{1, \ldots, d\}$. Compute the critical number $C = C(\pi)$, defined as the number of component failures required for the system to fail (that is, the system fails at the Cth component failure, for the given π). Note that C can also be computed by starting with all components failed and resurrecting them one by one in reverse order of their failure, until the system becomes operational. Computing C using this reverse order is often more efficient (Botev et al., 2016). Then compute the conditional cdf $\mathbb{P}[X \leq x \mid \pi]$, where X is the time of the Cth component failure. This is an unbiased estimator of $F(x)$ with smaller variance than the indicator $\mathbb{I}[X \leq x]$. It can also be shown that in an asymptotic regime in which the component failure rates converge to 0 so that $1 - F(x) \to 0$, the relative variance of this CMC estimator of $1 - F(x)$ remains bounded whereas it goes to infinity with the conventional estimator $\mathbb{I}[X > x]$; i.e., the CMC estimator has bounded relative error (Botev et al., 2013, 2016).

When the lifetimes are independent and exponential, X is a sum of C independent exponentials, so it has a hypoexponential distribution, whose cdf has an explicit formula that can be written in terms of a matrix exponential, and developed explicitly as a sum of products in terms of the rates of the exponential lifetimes, as explained in Botev et al. (2016). By taking the derivative of the conditional cdf formula with respect to x, one obtains the conditional density.

More specifically, let component j have an exponential lifetime with rate $\lambda_j > 0$, for $j = 1, \ldots, d$. For a given realization, let $\pi(j)$ be the jth component that fails and let $C(\pi) = c$ for the given π, let A_1 be the time until the first failure, and let A_j be the time between the $(j-1)$th and jth failures, for $j > 1$. Conditional on π, we have $X = A_1 + \cdots + A_c$ where the A_j's are independent and A_j is exponential with rate Λ_j for all $j \geq 1$, with $\Lambda_1 = \lambda_1 + \cdots + \lambda_d$, and $\Lambda_j = \Lambda_{j-1} - \lambda_{\pi(j)}$ for all $j \geq 2$. The conditional distribution of X is then hypoexponential with cdf

$$\mathbb{P}[X \leq x \mid \pi] = \mathbb{P}[A_1 + \cdots + A_c \leq x \mid \pi] = 1 - \sum_{j=1}^{c} p_j e^{-\Lambda_j x},$$

where

$$p_j = \prod_{k=1,k\neq j}^{c} \frac{\Lambda_k}{\Lambda_k - \Lambda_j}.$$

See Gertsbakh and Shpungin (2010), Appendix A, and Botev et al. (2016), for example. Taking the derivative with respect to x gives the CDE

$$f(x \mid \pi) = \sum_{j=1}^{c} \Lambda_j p_j e^{-\Lambda_j x},$$

in which c, the Λ_j and the p_j depend on π. This conditional density is well defined and computable everywhere in $[0,\infty)$. There are instability issues for computing p_j when $\Lambda_k - \Lambda_j$ is close to 0 for some $k \neq j$, but this can be addressed by a stable numerical algorithm of Higham (2009). All of this can be generalized easily to a model in which the lifetimes are dependent, with the dependence modeled by a Marshall-Olkin copula (Botev et al., 2016). In that model, the Y_j represent the occurrence times of shocks that can take down one or more components simultaneously.

It is interesting to note that although $f(x \mid \pi)$ is an unbiased estimator of the density $f(x)$ at any x, this estimator is a function of the permutation π only, so it takes its values in a finite set, which means that the corresponding $\tilde{g}(\mathbf{u})$ is a piecewise constant function, which is not RQMC-friendly. Therefore, we do not expect RQMC to bring a very large gain.

	$\hat{\nu}$	e19
MC	1.00	19.9
Lat+s	1.22	23.9
Lat+s+b	1.19	23.8
Sob+LMS	1.33	23.9

Table 4.7. Values of $\hat{\nu}$ and e19 with the CDE, for the network reliability example.

For a numerical illustration, we take the same graph as in Section 4.4.5. For $j = 1, \ldots, 13$, Y_j is exponential with rate λ_j and the Y_j are independent. The system fails as soon as there is no path going from 0 to 8. For simplicity, here we take $\lambda_j = 1$ for all j, although taking different λ_j's brings no significant additional difficulty. We estimate the density over the interval $(a,b] = (0, 1.829]$, which cuts off roughly 1% of the probability on the right side. Table 4.7 and Figure 4.7 give the results. The density of X estimated with $n = 2^{20}$ random samples is shown on the left and the IV plots are on the right. Despite the discontinuity of $\tilde{g}$, RQMC outperforms MC in terms of the IV by a factor of about $2^4 = 16$ for $n = 2^{19}$, and also by improving the empirical rate $\hat{\nu}$ to about -1.2 for lattices and even better with

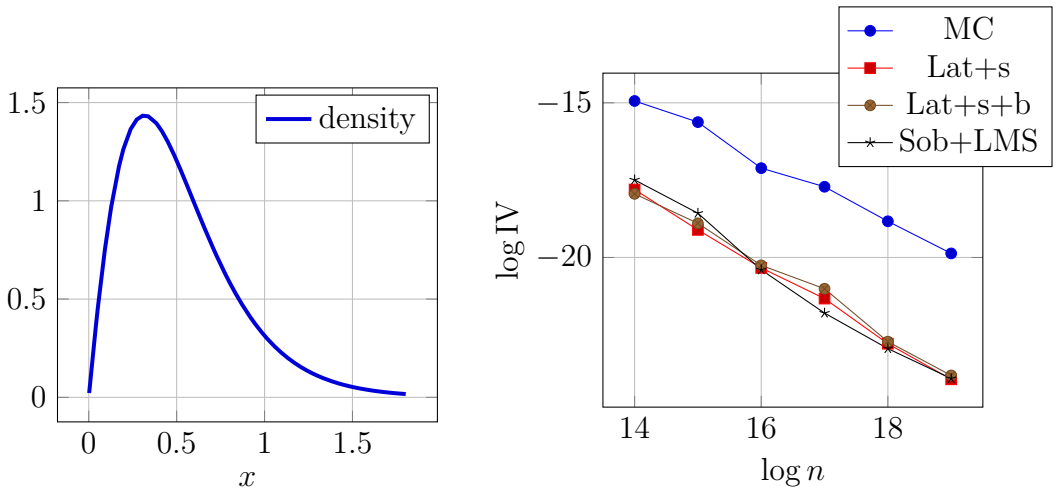

Figure 4.7. Density (left) and $\log \mathrm{IV}$ as a function of $\log n$ (right) for the network failure time.

Sobol' points. The Sobol' points used here were constructed using LatNet Builder (Marion et al., 2020) with a CBC search based on the t-value of all projections up to order 6, with order-dependent weights $\gamma_k = 0.8^k$ for projections of order k.

4.4.7. Density of waiting times in a single queue

4.4.7.1. *Model with independent days.*

We consider a single-server FIFO queue in which customers arrive from an arbitrary arrival process (not necessarily stationary Poisson) and the service times are independent, with continuous cdf G and density g. If W denotes the waiting time of a "random" customer, we want to estimate $p_0 = \mathbb{P}[W = 0]$ and the density f of W over $(0,\infty)$.

We first consider a system that starts empty and evolves over a fixed time horizon τ, which we call a *day*. Let T_j be the arrival time of the jth customer, $T_0 = 0$, $A_j = T_j - T_{j-1}$ the jth interarrival time, S_j the service time of customer j, and W_j the waiting time of customer j. Since the system starts empty, we have $W_1 = 0$, and the Lindley recurrence gives us that $W_j = \max(0, W_{j-1} + S_{j-1} - A_j)$ for $j \geq 2$. At time τ, the arrival process stops, but service continues until all customers already arrived are served. The number of customers handled in a day is the random variable $N = \max\{j \geq 1 : T_j < \tau\}$. The cdf of W can be written as $F(0) = p_0$ and for $x > 0$, $F(x) = \mathbb{P}[W \leq x] = \mathbb{E}[\mathbb{I}(W \leq x)]$. Note that the sequence of waiting times of all customers over an infinite number of independent successive

days is a regenerative process that regenerates at the beginning of each day. We can then apply the renewal reward theorem, which gives

$$F(x) = \mathbb{E}[\mathbb{I}(W \le x)] = \frac{\mathbb{E}\left[\mathbb{I}[W_1 \le x] + \cdots + \mathbb{I}[W_N \le x]\right]}{\mathbb{E}[N]}. \tag{4.4.4}$$

Since $\mathbb{E}[N]$ does not depend on x, we see that for $x > 0$, the density $f(x)$ is the derivative of the numerator with respect to x, divided by $\mathbb{E}[N]$.

To obtain a differentiable cdf estimator, we want to replace each indicator in the numerator by a conditional expectation. One simple way of doing this is to hide the service time S_{j-1} of the previous customer; that is, replace $\mathbb{I}[W_j \le x]$ by

$$P_j(x) = \mathbb{P}[W_j \le x \mid W_{j-1} - A_j] = \mathbb{P}[S_{j-1} \le x + A_j - W_{j-1}] = G(x + A_j - W_{j-1}) \quad \text{for } x \ge 0.$$

This gives $P_j(0) = G(A_j - W_{j-1})$ (there is a probability mass at 0), whereas for $x > 0$, we have $P_j'(x) = \mathrm{d}P_j(x)/\mathrm{d}x = g(x + A_j - W_{j-1})$ and then, since N does not change when we change x,

$$f(x) = \frac{\mathbb{E}[D(x)]}{\mathbb{E}[N]} \qquad \text{where } D(x) = \sum_{j=1}^{N} g(x + A_j - W_{j-1}). \tag{4.4.5}$$

Note that we are not conditioning on the same information for all terms of the sum, so what we do is not exactly CMC, but *extended CMC*. It nevertheless provides the required smoothing and an unbiased density estimator for the numerator of (4.4.4).

Often, for example if the arrival process is Poisson, $\mathbb{E}[N]$ can be computed exactly, in which case we only need to estimate $\mathbb{E}[D(x)]$ and we get an unbiased density estimator. Otherwise, the denominator $\mathbb{E}[N]$ can be estimated in the usual way. We are then in the standard setting of estimating a ratio of expectations (Asmussen and Glynn, 2007), for which we have unbiased estimators for the numerator and the denominator. We simulate n days, independently (with MC) or with n RQMC points, to obtain n realizations of $(N, D(x))$, say $(N_1, D_1(x)), \ldots, (N_n, D_n(x))$. The ratio estimator (CDE) of $f(x)$ is

$$\hat{f}(x) = \frac{\sum_{i=1}^{n} D_i(x)}{\sum_{i=1}^{n} N_i}. \tag{4.4.6}$$

It can be computed at any $x \in [0,\infty)$. For independent realizations (with MC), the variance of $\hat{f}(x)$ can be estimated using the delta method for ratio estimators (Asmussen and Glynn,

2007):

$$n\mathrm{Var}[\hat{f}(x)] \to \frac{\mathrm{Var}[D_i(x)] + \mathrm{Var}[N_i]f^2(x) - 2\mathrm{Cov}[D_i(x), N_i]f(x)}{\mathbb{E}^2[N_i]}$$

asymptotically, when $n \to \infty$. This variance can be estimated by replacing the unknown quantities in this expression by their empirical values. This is consistent because the n pairs $(D_i(x), N_i)$, $i = 1, \ldots, n$, are independent. Alternatively, a confidence interval on $f(x)$ can also be computed with a bootstrap approach (Choquet et al., 1999).

In the RQMC case, the pairs $(D_i(x), N_i)$ are no longer independent. Then, to obtain an estimator of $f(x)$ for which we can estimate the variance, we make n_r independent replicates of the RQMC estimator of the pair $(\mathbb{E}[D(x)], \mathbb{E}[N])$, say $(\bar{D}_1(x), \bar{N}_1), \ldots, (\bar{D}_{n_r}(x), \bar{N}_{n_r})$, where each $(\bar{D}_j(x), \bar{N}_j)$ is the average of n pairs $(D_i(x), N_i)$ sampled by RQMC. We estimate the density $f(x)$ by the ratio of the two grand sums

$$\hat{f}_{\mathrm{rqmc},n_r}(x) = \frac{\sum_{j=1}^{n_r} \bar{D}_j(x)}{\sum_{j=1}^{n_r} \bar{N}_j}.$$

To estimate the variance, we use that

$$\mathrm{Var}[\hat{f}_{\mathrm{rqmc},n_r}(x)] \approx \frac{\mathrm{Var}[\bar{D}_j(x)] + \mathrm{Var}[\bar{N}_j]f^2(x) - 2\mathrm{Cov}[\bar{D}_j(x), \bar{N}_j]f(x)}{n_r(\mathbb{E}[N])^2}$$

and we replace all the unknown quantities in this expression by their empirical values.

Here, the required dimension of the RQMC points is the (random) total number of inter-arrival times A_j and service times S_j that we need to generate during the day. It is approximately twice the number of customers that arrive during the day. This number is unbounded, so the RQMC points must have unbounded (or infinite) dimension, and one must be able to generate the points with-out first selecting a maximal dimension. Recurrence-based RQMC point sets have this property; they can be provided for instance by ordinary or polynomial Korobov lattice rules (L'Ecuyer and Lemieux, 1999, 2000, 2002; Lemieux and L'Ecuyer, 2003), which are available in the hups package of SSJ (L'Ecuyer, 2016).

4.4.7.2. *Steady-state model.*

In a slightly different setting, we can assume that the single queue evolves in steady-state over an infinite time horizon, under the additional assumptions that the A_j's are i.i.d. and the S_j's are also i.i.d. Again, we want to estimate the density of the waiting time W of a random customer. In this case, the system regenerates whenever a new customer arrives

in an empty system. The regenerative cycles can be much shorter on average than for the previous case, unless the day is very short or the utilization factor of the system is close to 1. The CDE has exactly the same form, apart from the different definition of regenerative cycle. In this case n represents the number of regenerative cycles, N_i is the number of customers in the ith cycle and $D_i(x)$ is the realization of $D(x)$ over the ith cycle.

In both settings, one could also hide A_j instead of S_{j-1}. The density estimator is similar and easy to derive. Intuition says that this should be a better choice if A_j has more variance than S_{j-1}.

4.4.7.3. *The GLRDE estimator*

Peng et al. (2020), Section 4.2.2., show how to construct a GLRDE estimator for the density of the *sojourn time* of customer j in this single-queue model. If the service times S_j are lognormal with parameters (μ, σ^2), we can write

$$X = W_j = \max(0, W_{j-1} + S_{j-1} - A_j) = \max(0, W_{j-1} + \exp[\sigma Z_{j-1} + \mu] - A_j) =: h(\mathbf{Y})$$

where Z_{j-1} has the standard normal density ϕ, and $\mathbf{Y} = (Y_1, Y_2, Y_3) = (Z_{j-1}, A_j, W_{j-1})$. When $W_j > 0$, taking the derivative of h with respect to $Y_1 = Z_{j-1}$ gives $h_1(\mathbf{Y}) = \exp[\sigma Z_{j-1} + \mu]\sigma = S_{j-1}\sigma$, $h_{11}(\mathbf{Y}) = S_{j-1}\sigma^2$, and these derivatives are 0 when $W_j = 0$. We also have $\partial \log \phi(x)/\partial x = -x$, and therefore for $x > 0$, $f(x) = \mathbb{E}[L(x)]/\mathbb{E}[N]$ where $L(x) = \sum_{j=1}^{N} \mathbb{I}[W_j \le x] \cdot \Psi_j$ and $\Psi_j = -(Z_{j-1} + \sigma)/(S_{j-1}\sigma)$. We can do n runs to estimate each of the two expectations in the ratio. This provides a very similar density estimator as with the CDE in (4.4.5), but here $L(x)$ is discontinuous in x, whereas $D(x)$ in (4.4.5) is continuous.

4.4.7.4. *Numerical results.*

For a numerical illustration, let the arrival process be Poisson with constant rate $\lambda = 1$, and the service times S_j lognormal with parameters $(\mu, \sigma^2) = (-0.7, 0.4)$. This gives $\mathbb{E}[S_j] = e^{-0.5} \approx 0.6065$ and $\mathrm{Var}[S_j] = e^{-1}(e^{0.4} - 1) \approx 0.18093$. For RQMC, we use infinite-dimensional RQMC points defined by Korobov lattice rules (L'Ecuyer and Lemieux, 2000) selected with `Lattice Builder` (L'Ecuyer and Munger, 2016) using order-dependent weights $\gamma_k = 0.005^k$ for projections of order k. We do not use Sobol' points because although they

can be constructed in an unlimited number of dimensions, with the available software the dimension must be fixed before generating the points and running the simulations.

Finite-horizon case. For the finite-horizon case, take $\tau = 60$, so $\mathbb{E}[N] = 60$, we only need toestimate the numerator, and we have an unbiased density estimator all over $[0,\infty)$. The results for $(a,b] = (0,2.2]$ are in Table 4.8 and Figure 4.8. An important observation is that CDE+MC provides an unbiased density estimator all over $[0,\infty)$. Due to the large and random dimensionality of the required RQMC points, and more importantly the discontinuity of the derivative of the estimator with respect to the underlying uniforms (because of the max, the HK variation is infinite), it was unclear if RQMC could bring any significant gain for this example. We see that although RQMC does not improve $\tilde{\nu}$ significantly, it improves the IV itself by a factor of about $2^{8.5} \approx 180$ for $n = 2^{19}$, which is quite significant. We also see that CDE beats GLRDE by a factor of about 500 with MC and about 200 with RQMC.

Steady-state case. We performed a similar experiment using regenerative simulation for the steady-state model. The density is similar but not exactly the same as in the finite-horizon case. The results are in Table 4.9 and Figure 4.9. They are very similar to those of the finite-horizon case, with similar empirical convergence rates, and the IV for $n = 2^{19}$ is again about 180 times smaller with CDE+RQMC compared to CDE+MC. The IV for GLRDE with $n = 2^{19}$ is roughly 1000 times larger than with CDE with MC and 250 times larger than with CDE with RQMC. The only important difference is that here, the IV is about 30 times *larger* than in the finite-horizon case, for all the methods. The explanation is that in the finite-horizon case, we simulate n runs with about 60 customers per run, whereas in the steady-state case, we have about 2.5 customers per regenerative cycle on average, so we simulate about 25 times fewer customers. Interestingly, the fact that we use much more coordinates of the RQMC points in the finite-horizon case (on average) makes no significant difference. A similar observation was made by L'Ecuyer and Lemieux (2000), Section 10.3, who compared finite-horizon runs of 5000 customers each on average, with regenerative simulation, in the context of estimating the probability of a large waiting time using RQMC. The reason why RQMC performs well even for a very large time horizon is that the integrand has low effective dimension in the successive-dimensions sense (L'Ecuyer and Lemieux, 2000).

		$\hat{\nu}$	e19
CDE	MC	1.00	24.8
	Lat+s	0.99	32.3
	Lat+s+b	1.02	32.3
GLRDE	MC	1.00	15.8
	Lat+s	1.01	24.6
	Lat+s+b	1.03	24.7

Table 4.8. Values of $\hat{\nu}$ and e19 for the single queue example, finite-horizon case.

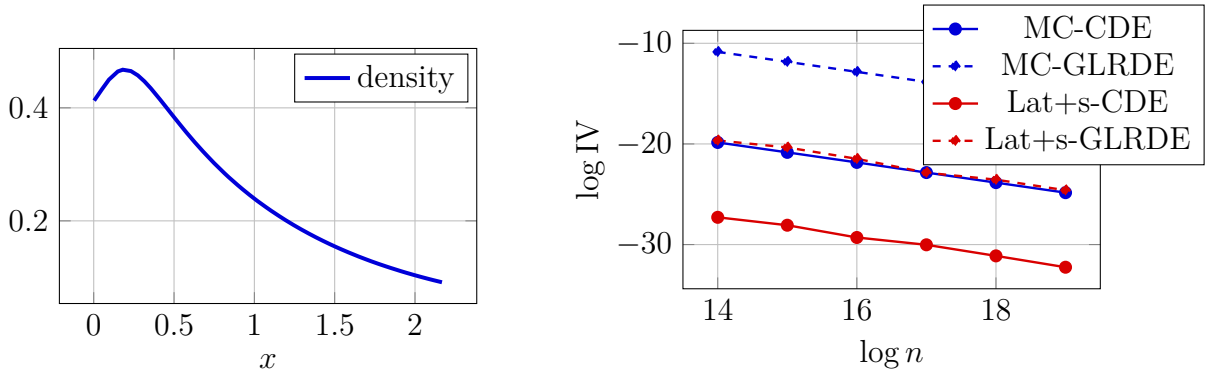

Figure 4.8. Estimated density (left) and $\log \mathrm{IV}$ as a function of $\log n$ (right) for the single queue over a finite-horizon.

		$\hat{\nu}$	e19
CDE	MC	0.99	19.9
	Lat+s	1.04	27.6
	Lat+s+b	1.08	27.8
GLR	MC	0.99	11.5
	Lat+s	1.20	20.1
	Lat+s+b	1.21	20.4

Table 4.9. Values of $\hat{\nu}$ and e19 for the single queue example, steady-state case.

4.4.8. A change of variable

In many situations, $X = h(\mathbf{Y})$ for a random vector $\mathbf{Y}$ and hiding a single coordinate of $\mathbf{Y}$ does not provide a very effective CDE. But sometimes, after an appropriate change of variable $\mathbf{Y} = g(\mathbf{Z})$, hiding one coordinate of the random vector $\mathbf{Z}$ can provide a much more effective CDE. Specifically, let $\mathbf{Z}_{-j}$ denote the vector $\mathbf{Z}$ with Z_j (the jth coordinate) removed, and let $\gamma(z) = \gamma(z; \mathbf{Z}_{-j}) = h(g(z; \mathbf{Z}_{-j}))$ denote the value of $h(\mathbf{Y})$ as a function of $Z_j = z$ when $\mathbf{Z}_{-j}$ is fixed. We assume in the following that for almost any realization of $\mathbf{Z}_{-j}$, $\gamma(z; \mathbf{Z}_{-j})$ is a monotone non-decreasing and differentiable function of z, so that

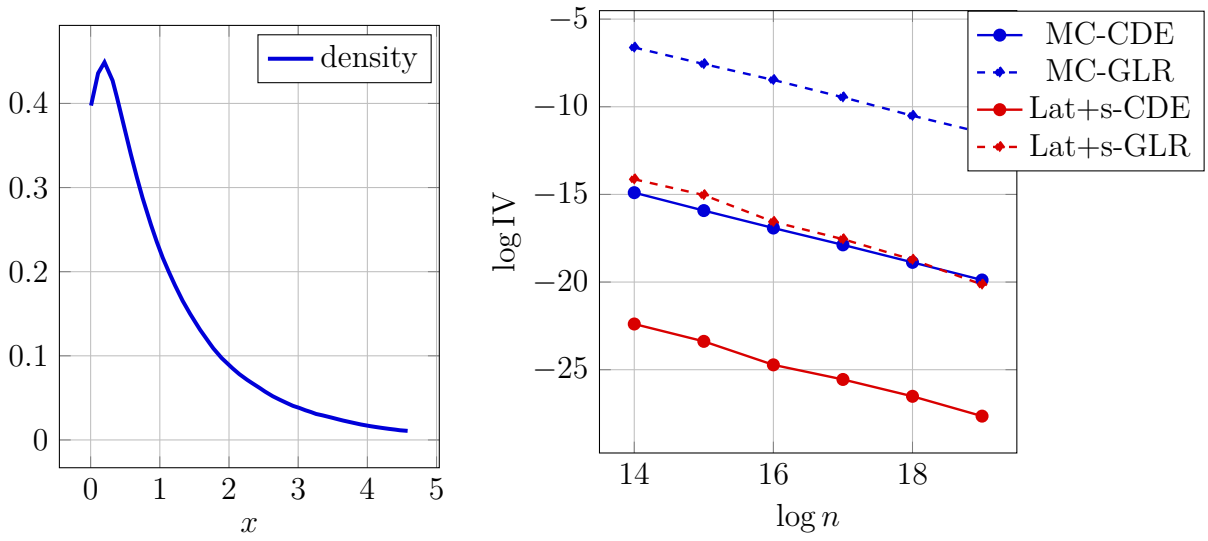

Figure 4.9. Estimated density (left) and $\log \mathrm{IV}$ as a function of $\log n$ (right) for the single queue in steady-state.

$\gamma^{-1}(x) = \inf\{z \in \mathbb{R} : \gamma(z) \geq x\}$ is well defined for any x. We also assume that Z_j has density ϕ and is independent of $\mathbf{Z}_{-j}$ (to simplify). Conditional on $\mathbf{Z}_{-j}$, we have

$$\mathbb{P}[x < h(\mathbf{Y}) \leq x+\delta \mid \mathbf{Z}_{-j}] = \mathbb{P}[x < \gamma(Z_j) \leq x+\delta \mid \mathbf{Z}_{-j}] = \mathbb{P}[z < Z_j \leq z+\Delta \mid \mathbf{Z}_{-j}] \approx \phi(z)\Delta$$

where $z = \gamma^{-1}(x)$, $z+\Delta = \gamma^{-1}(x+\delta)$. Taking the limit gives

$$f(x \mid \mathbf{Z}_{-j}) = \lim_{\delta \to 0} \frac{\mathbb{P}[z < Z_j \leq z+\Delta \mid \mathbf{Z}_{-j}]}{\delta} = \lim_{\delta \to 0} \frac{\phi(z)\Delta}{\delta} = \frac{\phi(z)}{\gamma'(z)} = \frac{\phi(\gamma^{-1}(x))}{\gamma'(\gamma^{-1}(x))}.$$

assuming that the latter is well defined. In case there are closed-form formulas for γ^{-1} and γ', this CDE can be evaluated directly. Otherwise, $z = \gamma^{-1}(x)$ can often be computed by a few iterations of a root-finding algorithm. Since γ and its inverse γ^{-1} depend on $\mathbf{Z}_{-j}$, this could mean inverting a different function for each sample realization. Our next example will show that the approach could nevertheless bring a huge benefit.

4.4.9. A function of a multivariate normal vector

We consider a multivariate normal vector $\mathbf{Y} = (Y_1, \cdots, Y_s)^{\mathsf{t}}$ defined via $Y_j = Y_{j-1} + \mu_j + \sigma_j Z_j$ with $Y_0 = 0$, the μ_j and $\sigma_j > 0$ are constants, and the Z_j are independent $\mathcal{N}(0,1)$ random variables, with cdf Φ and density ϕ. Let $X = \bar{S} = (S_1 + \cdots + S_s)/s$ where $S_j = S_0 e^{Y_j}$ for some constant $S_0 > 0$. We want to estimate the density of X over some

interval $(a,b) = (K,K+c)$ where $K \geq 0$ and $c > 0$. This is the same as estimating the density of $\max(0, \bar{S} - K)$, which may represent the payoff of some financial contract, for example (Glasserman, 2004). A simple way to define the CDE here is to hide Z_s. The conditional cdf is $\mathbb{P}[X \leq x | Z_s] = \mathbb{P}[Z_s \leq W(x)] = \Psi(W(x))$ where

$$W(x) = (\ln[sx - (S_1 + \cdots + S_{s-1})] - \ln S_0 - Y_{s-1} - \mu_j)/\sigma_j.$$

Taking the derivative with respect to x gives the unbiased CDE

$$f(x \mid \mathbf{Z}_{-j}) = \frac{\partial}{\partial x}\mathbb{P}[\bar{S} \leq x \mid \mathbf{Z}_{-s}] \;=\; \phi(W(x))W'(x) \;=\; \frac{\phi(W(x))s}{[sx - (S_1 + \cdots + S_{s-1})/S_0]\sigma_j} \quad (4.4.7)$$

Unfortunately, this *sequential* CDE is ineffective, because hiding only this Z_s does not remove much information, and then the conditional density has a large variance.

We now describe a less obvious but more effective conditioning approach. The goal is to hide a variable that contains more information. For this, we generate the vector $\mathbf{Y}$ using a Brownian bridge construction in which the Z_j's are used in a different way, as follows (Glasserman, 2004). Let $\bar{\mu} = \mu_1 + \cdots + \mu_j$ and $\bar{\sigma} = \sigma_1 + \cdots + \sigma_j$, for $j = 1, \ldots, s$. With this construction, we first sample $Y_s = \bar{\mu}_s + \bar{\sigma}_s Z_s$. Then, given $Y_s = y_s$, we put $j_2 = \lfloor s/2 \rfloor$ and we sample Y_{j_2} from its normal distribution conditional on $Y_s = y_s$, which is normal with mean $y_s \bar{\mu}_{j_2}/\bar{\mu}_s$ and variance $(\bar{\sigma}_s - \bar{\sigma}_{j_2})\bar{\sigma}_{j_2}/\bar{\sigma}_s$. This uses the fact that if X_1 and X_2 are independent and normal, then conditional on $X_1 + X_2 = \bar{x}$, X_1 is normal with mean $\bar{x}\mathbb{E}[X_1]/\mathbb{E}[X_1 + X_2]$ and variance $\mathrm{Var}[X_1]\mathrm{Var}[X_2]/\mathrm{Var}[X_1 + X_2]$. Then we put $j_3 = \lfloor j_2/2 \rfloor$ and we sample Y_{j_3} conditionally on Y_{j_2}, then we put $j_4 = \lfloor (j_2 + s)/2 \rfloor$ and we sample Y_{j_4} conditionally on (Y_{j_2}, Y_s), and so on, until all the Y_j's are known.

For the CDE, we can hide again Z_s, but now Z_s has much more impact on the payoff, because all the Y_j's depend on Z_s. This makes the conditional density muchless straightforward to compute, but we can proceed as follows. To avoid sampling Z_s, we sample $Y_1, \ldots, Y_{s-1}$ conditional on $Z_s = z_s = 0$, which will give say $Y_1^0, \ldots, Y_{s-1}^0$, and then write X as a function of $z = z_s$ conditional on these values, that is, conditional on $\mathbf{Z}_{-s} = (Z_1, \ldots, Z_{s-1})$. We have $Y_s = Y_s^0 + \bar{\sigma}_s Z_s$ and $Y_j = Y_j^0 + (\bar{\mu}_j/\bar{\mu}_s)\bar{\sigma}_s Z_s$. Then,

$$X = \bar{S} = \frac{S_0}{s}\sum_{j=1}^{s} e^{Y_j} = \frac{S_0}{s}\sum_{j=1}^{s} \exp[Y_j^0 + Z_s(\bar{\mu}_j/\bar{\mu}_s)\bar{\sigma}_s].$$

This fits the framework of Section 4.4.8, with $j = s$,

$$\gamma(z) = \frac{S_0}{s} \sum_{j=1}^{s} \exp[Y_j^0 + z(\bar{\mu}_j/\bar{\mu}_s)\bar{\sigma}_s].$$

and

$$\gamma'(z) = \frac{S_0}{s} \sum_{j=1}^{s} \exp[Y_j^0 + z(\bar{\mu}_j/\bar{\mu}_s)\bar{\sigma}_s](\bar{\mu}_j/\bar{\mu}_s)\bar{\sigma}_s.$$

The CDE at $x = \gamma(z)$ is then $f(x \mid \mathbf{Z}_{-j}) = \phi(z)/\gamma'(z)$. We call it the *bridge* CDE.

To compute this density at a specified x, we need to compute $z = \gamma^{-1}(x)$. We have no explicit formula for γ^{-1} in this case, but we can compute a root of $\gamma(z) - x = 0$ numerically. To evaluate the density at the n_e evaluation points $e_1, \ldots, e_{n_e}$ in (a,b), we can proceed as follows. We first compute $x_* = \gamma(0)$ and let j_* be the smallest j for which $e_j \geq x_*$. We compute $z = w_{j_*}$ such that $\gamma(w_{j_*}) = e_{j_*}$. This can be done via Newton iteration, $z_k = z_{k-1} - (\gamma(z_{k-1}) - e_{j^*})/\gamma'(z_{k-1})$, starting with $z_0 = 0$. Then, for $j = j_* + 1, \ldots, n_e$, we use again Newton iteration to find $z = w_j$ such that $\gamma(w_j) = e_j$, starting at $z_0 = w_{j-1}$. We do the same to find $z = w_j$ such that $\gamma(w_j) = e_j$ for $j = j_* - 1, \ldots, 1$, starting at $z_0 = w_{j+1}$. This provides the point w_j required to evaluate the conditional density at e_j, for each j. We have to repeat this procedure for each realization of $\mathbf{Z}_{-j}$, because the function γ depends on $\mathbf{Z}_{-j}$. This implies additional computations. However, the gain in accuracy can be quite significant.

		$\hat{\nu}$	e19
sequential KDE	MC	-0.78	-20.4
	Sob+LMS	-0.76	-20.6
sequential CDE	MC	-1.00	-19.9
	Lat+s	-1.07	-20.3
	Lat+s+b	-1.01	-20.1
	Sob+LMS	-1.00	-20.0
bridge CDE	MC	-1.04	-27.9
	Lat+s	-1.60	-40.0
	Lat+s+b	-1.74	-45.0
	Sob+LMS	-2.01	-46.9

Table 4.10. Values of $\hat{\nu}$ and e19 for the Asian option, with sequential and bridge CDE constructions.

For a numerical illustration, we take $S_0 = 100$, $s = 12$, $r = 0.1$, $\mu_j = 0.00771966$, $\sigma_j = 0.035033$, $K = 101$. We estimate the density over $[a,b] = [101, 128.13]$. To approximate the root of $\gamma(z) - x = 0$ for the bridge CDE, we use five Newton iterations; doing more

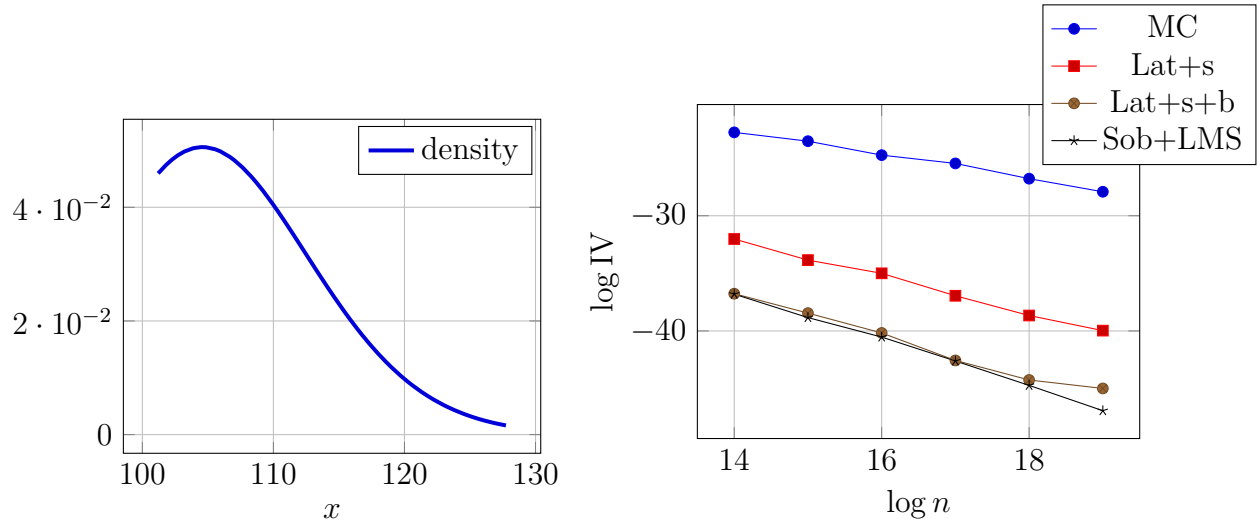

Figure 4.10. Estimated density (left) and log IV as a function of log n (left) for the Asian option.

iterations made no significant difference. The results are in Table 4.10 and Figure 4.10. We find that RQMC with the bridge CDE performs extremely well. For example, for Sob+LMS, the MISE with $n = 2^{19}$ is approximately $2^{-46.9}$, which is about 2^{19} (half a million) times smaller than for the same CDE with MC, and it decreases as n^{-2}. With a KDE, the MISE with $n = 2^{19}$ is about $2^{21} \approx 2$ million times larger with the same Sobol' points and $2^{26} \approx 67$ million times larger with MC. With the sequential CDE, RQMC is ineffective and the IV of the MC estimator is also quite large, as expected.

To illustrate the behavior of the sequential and bridge CDEs, Figure 4.11 plots 5 single realizations of each, the same horizontal scale. The sequential CDE has much more spiky realizations than the bridge CDE, and this explains why the latter performs much better.

4.4.10. Estimating a quantile with a confidence interval

For $0 < q < 1$, the q-quantile of the distribution of X is defined as $\xi_q = F^{-1}(q) = \inf\{x : F(x) \geq q\}$. Given n i.i.d. observations of X, a standard (consistent) estimator of ξ_q is the q-quantile of the empirical distribution, defined as $\hat{\xi}_{q,n} = X_{(\lceil nq \rceil)}$, where $X_{(1)}, \ldots, X_{(n)}$ are the n observations sorted in increasing order (the order statistics). We assume that the density $f(x)$ is positive and continuously differentiable in a neighborhood of ξ_q. Then we have the central limit theorem (CLT):

$$\sqrt{n}(\hat{\xi}_{q,n} - \xi_q)/\sigma_\xi \Rightarrow \mathcal{N}(0,1) \qquad \text{for } n \to \infty,$$

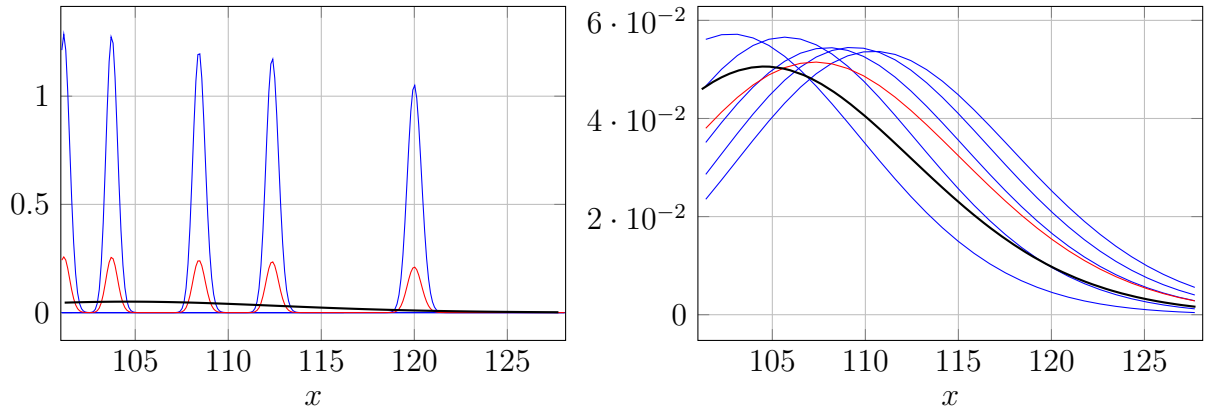

Figure 4.11. Five realizations of the density estimator (blue), their average (red), and the true density (thick black) for the sequential CDE (left) and the bridge CDE (right), for the Asian option example.

where $\sigma_\xi^2 = q(1-q)/f^2(\xi_q)$ (Serfling, 1980). This provides a way to compute a confidence interval on ξ_q, but requires the estimation of $f(\xi_q)$, which is generally difficult. Some approaches for doing this include finite differences with the empirical cdf, batching, and sectioning (Asmussen and Glynn, 2007; Nakayama, 2014a,b).

In our setting, one can do better by taking the q-quantile $\hat{\xi}_{\text{cmc},q,n}$ of the conditional cdf

$$\hat{F}_{\text{cmc},n}(x) = \frac{1}{n}\sum_{i=1}^{n} F(x \mid \mathcal{G}^{(i)}).$$

That is, $\hat{\xi}_{\text{cmc},q,n} = \inf\{x : \hat{F}_{\text{cmc},n}(x) \geq q\}$. This idea was already suggested by Nakayama (2014b), who pointed out that this estimator obeys a CLT just like $\hat{\xi}_{q,n}$, but with the variance constant σ_ξ^2 replaced by $\sigma_{\text{cmc},\xi}^2 = \text{Var}[F(\xi_q \mid \mathcal{G})]/f^2(\xi_q) \leq \sigma_\xi^2$. This is an improvement on the quantile estimator itself. Our CDE approach also provides an improved estimator of the density $f(\xi_q)$ which appears in the variance expression. We estimate $f(\xi_q)$ by $\hat{f}_{\text{cde},n}(\hat{\xi}_{\text{cmc},q,n})$. This provides a more accurate confidence interval of ξ_q.

Further improvements on the variances of both the quantile and density estimators can be obtained by using RQMC to generate the realizations $\mathcal{G}^{(i)}$. In particular, if $\tilde{g}(\xi_q, \mathbf{u}) = F(\xi_q \mid \mathcal{G})$ is a sufficiently smooth function of $\mathbf{u}$, $\text{Var}[\hat{\xi}_{\text{cmc},q,n}]$ can converge at a faster rate than $\mathcal{O}(n^{-1})$. When using RQMC with n_r randomizations to estimate a quantile, the quantile estimator will be the empirical quantile of all the $n_r \times n$ observations.

A related quantity is the *expected shortfall*, defined as $c_q = \mathbb{E}[X \mid X > \xi_q] = \xi_q - \mathbb{E}[(\xi_q - X)^+]/q$ which is often estimated by its empirical version (Hong et al., 2014)

$$\hat{c}_{q,n} = \hat{\xi}_{q,n} - \frac{1}{nq}\sum_{i=1}^{n}(\hat{\xi}_{q,n} - X_i)^+.$$

This estimator obeys the CLT $\sqrt{n}(\hat{c}_{q,n} - c_q)/\sigma_c \Rightarrow \mathcal{N}(0,1)$ for $n \to \infty$, where $\sigma_c^2 = \mathrm{Var}[(\xi_q - X)^+]/q^2$, if this variance is finite (Hong et al., 2014). By improving the quantile estimator, CDE+RQMC can also improve the expected shortfall estimator a well as the estimator of the variance constant σ_c^2 and the quality of confidence intervals on c_q. We leave this as a topic for future work.

4.5. Conclusion

We have examined a simple and very effective approach for estimating the density of a random variable generated by simulation from a stochastic model, by conditioning. The resulting CDE is unbiased and its MISE converges at a faster rate than for other popular density estimators such as the KDE. We have also shown how to further reduce the IV, and even improve its convergence rate, by combining the CDE with RQMC sampling. Our numerical examples show that this combination can be very efficient. It sometimes reduces the MISE by factors over a million. Our CDE approach also outperforms the recently proposed GLRDE method, and CDE+RQMC outperforms both GLRDE+RQMC and KDE+RQMC, in all our examples.

Suggested future work includes experimenting this methodology on larger and more complicated stochastic models, designing and exploring different types of conditioning, and perhaps adapting the Monte Carlo sampling strategies to make the method more effective (e.g., by changing the way X is defined in terms the basic input random variates). Its application to quantile and expected shortfall estimation also deserves further study.

Acknowledgments

This work has been supported by an IVADO Research Grant, an NSERC-Canada Discovery Grant, a Canada Research Chair, and an Inria International Chair, to P. L'Ecuyer. F. Puchhammer was also supported by Spanish and Basque governments fundings through BCAM (ERDF, ESF, SEV-2017-0718, PID2019-108111RB-I00, PID2019-104927GB-C22,

BERC 2018e2021, EXP. 2019/00432, ELKARTEK KK-2020/00049), and the computing infrastructure of i2BASQUE academic network and IZO-SGI SGIker (UPV). We thank Julien Keutchayan for pointing out mistakes in a previous version.

Chapter 5

Article 3: Array-RQMC for option pricing under stochastic volatility models

In the third article, we show that Array-RQMC can be applied for option pricing under a stochastic volatility process such as the variance gamma, Heston, and Ornstein-Uhlenbeck models. We describe and compare different implementation alternatives, and report empirical experiments to determine the benefit of using this method compared to MC, even if it requires RQMC points in larger dimension.

This article has been published in the proceedings of the 2019 Winter Simulation Conference, IEEE Press. Preliminary work was presented at the following conferences:

— The Optimization Days, Montreal, May 2019;

— Winter Simulation Conference, National Harbor, Maryland, December 2019.

The main author contributions are:

— The general research ideas were proposed by Pierre L'Ecuyer;

— The research (including implementation, experiments, etc.) was carried out by Amal Ben Abdellah except for the idea of linear sort for the VG example, which it was suggested by Florian Puchhammer;

— The article was written by Amal Ben Abdellah, and it was revised and corrected by Pierre L'Ecuyer.

Array-RQMC for option pricing under stochastic volatility models

Amal Ben Abdellah, Pierre L'Ecuyer and Florian Puchhammer
Département d'Informatique et de Recherche Opérationnelle,
Pavillon Aisenstadt, Université de Montréal, C.P. 6128, Succ. Centre-Ville,
Montréal, Québec, Canada H3C 3J7.

Abstract

Array-RQMC has been proposed as a way to effectively apply randomized quasi-Monte Carlo (RQMC) when simulating a Markov chain over a large number of steps to estimate an expected cost or reward. The method can be very effective when the state of the chain has low dimension. For pricing an Asian option under an ordinary geometric Brownian motion model, for example, Array-RQMC can reduce the variance by factors in the millions. In this paper, we show how to apply this method and we study its effectiveness in case the underlying process has stochastic volatility. We show that Array-RQMC can also work very well for these models, even if it requires RQMC points in larger dimension. We examine in particular the variance-gamma, Heston, and Ornstein-Uhlenbeck stochastic volatility models, and we provide numerical results.

5.1. Introduction

Quasi-Monte Carlo (QMC) and randomized QMC (RQMC) methods can improve efficiency significantly when estimating an integral in a moderate number of dimensions, but their use for simulating Markov chains over a large number of steps has been limited so far. The array-RQMC method, developed for that purpose, has been shown to work well for some chains having a low-dimensional state. It simulates an array of n copies of the Markov chain so that each chain follows its exact distribution, but the copies are not independent, and the empirical distribution of the states at any given step of the chain is a "low-discrepancy" approximation of the exact distribution. At each step, the n chains (or states) are matched one-to-one to a set of n RQMC points whose dimension is the dimension of the state plus the number of uniform random numbers required to advance the chain by one more step. The first coordinates of the points are used to match the states to the points and the other

coordinates provide the random numbers needed to determine the next state. When the chains have a large-dimensional state, the dimension used for the match can be reduced via a mapping to a lower-dimensional space. Then the matching is performed by sorting both the points and the chains. When the dimension of the state exceeds 1, this matching is done via a multivariate sort. The main idea is to evolve the array of chains in a way that from step to step, the empirical distribution of the states keeps its low discrepancy. For further details on the methodology, sorting strategies, convergence analysis, applications, and empirical results, we refer the reader to Demers et al. (2005); Dion and L'Ecuyer (2010); El Haddad et al. (2008, 2010); Gerber and Chopin (2015); Lécot and Tuffin (2004); L'Ecuyer (2018); L'Ecuyer and Sanvido (2010); L'Ecuyer et al. (2006, 2007, 2008, 2009), and the other references given there.

The aim of this paper is to examine how Array-RQMC can be applied for option pricing under a stochastic volatility process such as the variance gamma, Heston, and Ornstein-Uhlenbeck models. We explain and compare various implementation alternatives, and report empirical experiments to assess the (possible) gain in efficiency and convergence rate. A second objective is for the WSC community to become better aware of this method, which can have numerous other applications.

Array-RQMC has already been applied for pricing Asian options when the underlying process evolves as a geometric Brownian motion (GBM) with fixed volatility L'Ecuyer (2018); L'Ecuyer et al. (2009). In that case, the state is two-dimensional (it contains the current value of the GBM and its running average) and a single random number is needed at each step, so the required RQMC points are three-dimensional. In their experiments, L'Ecuyer (2018) observed an empirical variance of the average payoff that decreased approximately as $\mathcal{O}(n^{-2})$ for Array-RQMC, in a range of reasonable values of n, compared with $\mathcal{O}(n^{-1})$ for independent random points (Monte Carlo). For $n = 2^{20}$ (about one million chains), the variance ratio between Monte Carlo and Array-RQMC was around 2 to 4 millions.

In view of this spectacular success, one wonders how well the method would perform when the underlying process is more involved, e.g., when it has stochastic volatility. This is relevant because stochastic volatility models are more realistic than the plain GBM model Madan and Seneta (1990); Madan et al. (1998). Success is not guaranteed because the dimension of the required RQMC points is larger. For the Heston model, for example, the

RQMC points must be five-dimensional instead of three-dimensional, because the state has three dimensions and we need two uniform random numbers at each step. It is unclear a priori if there will be any significant variance reduction for reasonable values of n.

The remainder is organized as follows. In Section 5.2, we state our general Markov chain model and provide background on the Array-RQMC algorithm, including matching and sorting strategies. In Section 5.3, we describe our experimental setting, and the types of RQMC point sets that we consider. Then we study the application of Array-RQMC under the variance-gamma model in Section 5.4, the Heston model in Section 5.5, and the Ornstein-Uhlenbeck model in Section 5.6. We end with a conclusion.

5.2. Background: Markov Chain Model, RQMC, and Array-RQMC

The option pricing models considered in this paper fit the following framework, which we use to summarize the Array-RQMC algorithm. We have a discrete-time *Markov chain* $\{X_j, j \geq 0\}$ defined by a stochastic recurrence over a measurable state space $\mathcal{X}$:

$$X_0 = x_0, \qquad \text{and} \qquad X_j = \varphi_j(X_{j-1}, \mathbf{U}_j), \quad j = 1, \ldots, \tau. \tag{5.2.1}$$

where $x_0 \in \mathcal{X}$ is a deterministic initial state, $\mathbf{U}_1, \mathbf{U}_2, \ldots$ are independent random vectors uniformly distributed over the d-dimensional unit cube $(0,1)^d$, the functions $\varphi_j : \mathcal{X} \times (0,1)^d \to \mathcal{X}$ are measurable, and τ is a fixed positive integer (the time horizon). The goal is:

$$\text{Estimate} \quad \mu_{\mathrm{y}} = \mathbb{E}[Y], \qquad \text{where} \quad Y = g(X_\tau)$$

and $g : \mathcal{X} \to \mathbb{R}$ is a *cost* (or reward) function. Here we have a cost only at the last step but in general there can be a cost function for each step and Y would be the sum of these costs L'Ecuyer et al. (2008).

Crude Monte Carlo estimates μ by the average $\bar{Y}_n = \frac{1}{n}\sum_{i=0}^{n-1} Y_i$, where $Y_0, \ldots, Y_{n-1}$ are n independent realizations of Y. One has $\mathbb{E}[\bar{Y}_n] = \mu_{\mathrm{y}}$ and $\mathrm{Var}[\bar{Y}_n] = \mathrm{Var}[Y]/n$, assuming that $\mathbb{E}[Y^2] = \sigma_{\mathrm{y}}^2 < \infty$. Note that the simulation of each realization of Y requires a vector $\mathbf{V} = (\mathbf{U}_1, \ldots, \mathbf{U}_\tau)$ of $d\tau$ independent uniform random variables over (0,1), and crude Monte Carlo produces n independent replicates of this random vector.

Randomized quasi-Monte Carlo (RQMC) replaces the n independent realizations of $\mathbf{V}$ by n *dependent* realizations, which form an RQMC point set in $d\tau$ dimensions. That is, each

$\mathbf{V}_i$ has the uniform distribution over $[0,1)^{d\tau}$, and the point set $P_n = \{V_0,...,V_{n-1}\}$ covers $[0,1)^{d\tau}$ more evenly than typical independent random points. With RQMC, $\bar{Y}_n$ remains an unbiased estimator of μ, but its variance can be much smaller, and can converge faster than $\mathcal{O}(1/n)$ under certain conditions. For more details, see Dick and Pillichshammer (2010); L'Ecuyer (2009, 2018); L'Ecuyer and Lemieux (2000), for example. However, when $d\tau$ is large, standard RQMC typically becomes ineffective, in the sense that it does not bring much variance reduction unless the problem has special structure.

Array-RQMC is an alternative approach developed specifically for Markov chains L'Ecuyer et al. (2006, 2008, 2018). To explain how it works, let us first suppose for simplicity (we will relax it later) that there is a mapping $h : \mathcal{X} \to \mathbb{R}$, that assigns to each state a *value* (or score) which summarizes in a single real number the most important information that we should retain from that state (like the value function in stochastic dynamic programming). This h is called the *sorting function*. The algorithm simulates n (dependent) realizations of the chain "in parallel". Let $X_{i,j}$ denote the state of chain i at step j, for $i = 0,\ldots,n-1$ and $j = 0,\ldots,\tau$. At step j, the n chains are sorted by increasing order of their values of $h(X_{i,j-1})$, the n points of an RQMC point set in $d+1$ dimensions are sorted by their first coordinate, and each point is matched to the chain having the same position in this ordering. Each chain i is then moved forward by one step, from state $X_{i,j-1}$ to state $X_{i,j}$, using the d other coordinates of its assigned RQMC point. Then we move on to the next step, the chains are sorted again, and so on.

The sorting function can in fact be more general and have the form $h : \mathcal{X} \to \mathbb{R}^c$ for some small integer $c \geq 1$. Then the mapping between the chains and the points must be realized in a c-dimensional space, i.e., via some kind of c-dimensional multivariate sort. The RQMC points then have $c+d$ coordinates, and are sorted with the same c-dimensional multivariate sort based on their first c coordinates, and mapped to the corresponding chains. The other d coordinates are used to move the chains ahead by one step. In practice, the first c coordinates of the RQMC points do not have to be randomized at each step; they are usually fixed and the points are already sorted in the correct order based on these coordinates.

Some multivariate sorts are described and compared by El Haddad et al. (2008); L'Ecuyer (2018); L'Ecuyer et al. (2009). For example, in a *multivariate batch sort*, we select positive integers $n_1,\ldots,n_c$ such that $n = n_1 \ldots n_c$. The states are first sorted by their first coordinate

in n_1 packets of size n/n_1, then each packet is sorted by the second coordinate into n_2 packets of size n/n_1n_2, and so on. The RQMC points are sorted in exactly the same way, based on their first c coordinates. In the *multivariate split sort*, we assume that $n = 2^e$ and we take $n_1 = n_2 = \cdots = n_e = 2$. That is, we first split the points in 2 packets based on the first coordinate, then split each packet in two by the second coordinate, and so on. If $e > c$, after c splits we get back to the first coordinate and continue.

Examples of heuristic sorting functions $h : \mathcal{X} \to \mathbb{R}$ are given in L'Ecuyer et al. (2008, 2018). Wächter and Keller (2008) and Gerber and Chopin (2015) suggested to first map the c-dimensional states to $[0,1]^c$ and then use a space filling curve in $[0,1]^c$ to map them to $[0,1]$, which provides a total order. Gerber and Chopin (2015) proposed to map the states to $[0,1]^c$ via a component-wise rescaled logistic transformation, then order them with a Hilbert space-filling curve. See L'Ecuyer (2018) for a more detailed discussion. Under smoothness conditions, they proved that the resulting unbiased Array-RQMC estimator has $o(1/n)$ variance, which beats the $\mathcal{O}(1/n)$ Monte Carlo rate.

Algorithm 2 states the Array-RQMC procedure in our setting. Indentation delimits the scope of the **"for"** loops. For any choice of sorting function h, the average $\hat{\mu}_{\text{arqmc},n} = \bar{Y}_n$ returned by this algorithm is always an *unbiased* estimator of μ. An unbiased estimator of $\text{Var}[\bar{Y}_n]$ can be obtained by making m independent realizations of $\hat{\mu}_{\text{arqmc},n}$ and computing their *empirical variance*.

Algorithm 2 : Array-RQMC Algorithm for Our Setting

for $i = 0, \ldots, n-1$ **do** $X_{i,0} \leftarrow x_0$;
for $j = 1, 2, \ldots, \tau$ **do**
  Sorting: Compute an appropriate permutation π_j of the n chains, based on the $h(X_{i,j-1})$, to match the n states with the RQMC points;
  Randomize afresh the RQMC points $\{\mathbf{U}_{0,j}, \ldots, \mathbf{U}_{n-1,j}\}$;
  for $i = 0, \ldots, n-1$ **do** $X_{i,j} = \varphi_j(X_{\pi_j(i),j-1}, \mathbf{U}_{i,j})$;
end forreturn the average $\hat{\mu}_{\text{arqmc},n} = \bar{Y}_n = (1/n)\sum_{i=0}^{n-1} g(X_{i,\tau})$ as an estimate of μ_{y}.

5.3. Experimental Setting

For all the option pricing examples in this paper, we have an asset price that evolves as a stochastic process $\{S(t), t \geq 0\}$ and a payoff that depends on the values of this process at fixed observation times $0 = t_0 < t_1 < t_2 < \ldots < t_c = T$. More specifically, for given constants

r (the interest rate) and K (the strike price), we consider an *European option* whose payoff is

$$Y = Y_{\mathrm{e}} = g(S(T)) = e^{-rT} \max(S(T) - K, 0)$$

and a discretely-observed *Asian option* whose payoff is

$$Y = Y_{\mathrm{a}} = g(\bar{S}) = e^{-rT} \max(\bar{S} - K, 0)$$

where $\bar{S} = (1/c) \sum_{j=1}^{c} S(t_j)$. In this second case, the running average $\bar{S}_j = (1/j) \sum_{\ell=1}^{j} S(t_\ell)$ must be kept in the state of the Markov chain. The information required for the evolution of $S(t)$ depends on the model and is given for each model in forthcoming sections. It must be maintained in the state. For the case where S is a plain GBM, the state of the Markov chain at step j can be taken as $X_j = (S(t_j), \bar{S}_j)$, a two-dimensional state, as was done in L'Ecuyer et al. (2009) and L'Ecuyer et al. (2008).

In our examples, the states are always multidimensional. To match them with the RQMC points, we will use a split sort, a batch sort, and a Hilbert-curve sort, and compare these alternatives. The Hilbert sort requires a transformation of the ℓ-dimensional states to the unit hypercube $[0,1]^\ell$. For this, we use a *logistic transformation* defined by $\psi(x) = (\psi_1(x_1),...,\psi_\ell(x_\ell)) \in [0,1]^\ell$ for all $x = (x_1, \ldots, x_\ell) \in \mathcal{X}$, where

$$\psi_j(x_j) = \left[1 + \exp\left(-\frac{x_j - \underline{x}_j}{\bar{x}_j - \underline{x}_j}\right)\right]^{-1}, \quad j = 1,...,\ell, \tag{5.3.1}$$

with constants $\bar{x}_j = \mu_j + 2\sigma_j$ and $\underline{x}_j = \mu_j - 2\sigma_j$ in which μ_j and σ_j are estimates of the mean and the variance of the distribution of the jth coordinate of the state. In Section 5.4, we will also consider just taking a linear combination of the two coordinates, to map a two-dimensional state to one dimension.

For RQMC, we consider

(1) Independent points, which corresponds to crude Monte Carlo (MC);

(2) Stratified sampling over the unit hypercube (Stratif);

(3) Sobol' points with a random linear matrix scrambling and a digital random shift (Sobol'+LMS);

(4) Sobol' points with nested uniform scrambling (Sobol'+NUS);

(5) A rank-1 lattice rule with a random shift modulo 1 followed by a baker's transformation (Lattice+baker).

The first two are not really RQMC points, but we use them for comparison. For stratified sampling, we divide the unit hypercube into $n = k^{\ell+d}$ congruent subcubes for some integer $k > 1$, and we draw one point randomly in each subcube. For a given target n, we take k as the integer for which $k^{\ell+d}$ is closest to this target n. For the Sobol' points, we took the default direction numbers in SSJ, which are from Lemieux et al. (2004). The LMS and NUS randomizations are explained in Owen (2003) and L'Ecuyer (2009). For the rank-1 lattice rules, we used generating vectors found by Lattice Builder (L'Ecuyer and Munger, 2016), using the $\mathcal{P}_2$ criterion with order-dependent weights $(0.8)^k$ for projections of order k.

For each example, each sorting method, each type of point set, and each selected value of n, we ran simulations to estimate $\mathrm{Var}[\bar{Y}_n]$. For the stratified and RQMC points, this variance was estimated by replicating the RQMC scheme $m = 100$ times independently. For a fair comparison with the MC variance $\sigma_{\mathrm{y}}^2 = \mathrm{Var}[Y]$, for these point sets we used the *variance per run*, defined as $n\mathrm{Var}[\bar{Y}_n]$. We define the *variance reduction factor* (VRF) for a given method compared with MC by $\sigma_{\mathrm{y}}^2/(n\mathrm{Var}[\bar{Y}_n])$. In each case, we fitted a linear regression model for the variance per run as a function of n, in log-log scale. We denote by $\hat{\beta}$ the regression slope estimated by this linear model.

In the remaining sections, we explain how the process $\{S(t), t \geq 0\}$ is defined in each case, how it is simulated. We show how we can apply Array-RQMC and we provide numerical results. All the experiments were done in Java using the SSJ library (L'Ecuyer, 2016; L'Ecuyer and Buist, 2005).

5.4. Option Pricing Under A Varaince-Gamma Process

The *variance-gamma* (VG) model was proposed for option pricing by Madan and Seneta (1990) and Madan et al. (1998), and further studied by Avramidis and L'Ecuyer (2006); Avramidis et al. (2003); Fu et al. (1998), for example. A VG process is essentially a Brownian process for which the time clock runs at random and time-varying speed driven by a gamma process. The VG process with parameters (θ,σ^2,ν) is defined as $Y = \{Y(t) = X(G(t)), t \geq 0\}$ where $X = \{X(t), t \geq 0\}$ is a Brownian motion with drift and variance parameters θ and σ^2, and $G = \{G(t), t \geq 0\}$ is a gamma process with drift and volatility parameters 1 and ν, independent of X. This means that $X(0) = 0$, $G(0) = 0$, both B and G have independent increments, and for all $t \geq 0$ and $\delta > 0$, we have $X(t+\delta) - X(t) \sim \mathrm{Normal}(\delta\theta,\delta\sigma^2)$, a normal

random variable with mean $\delta\theta$ and variance $\delta\sigma^2$, and $G(t+\delta) - G(t) \sim \text{Gamma}(\delta/\nu, \nu)$, a gamma random variable with mean δ and variance $\delta\nu$. The gamma process is always non-decreasing, which ensures that the time clock never goes backward. In the VG model for option pricing, the asset value follows the *geometric variance-gamma* (GVG) process $S = \{S(t), t \geq 0\}$ defined by

$$S(t) = S(0) \exp\left[(r+\omega)t + X(G(t))\right],$$

where $\omega = \ln(1 - \theta\nu - \sigma^2\nu/2)/\nu$.

To generate realizations of $\bar{S}$ for this process, we must generate $S(t_1), \ldots, S(t_\tau)$, and there are many ways of doing this. With Array-RQMC, we want to do it via a Markov chain with a low-dimensional state. The running average $\bar{S}_j$ must be part of the state, as well as sufficient information to generate the future of the path. A simple procedure for generating the path is to sample sequentially $G(t_1)$, then $Y(t_1) = X(G(t_1))$ conditional on $G(t_1)$, then $G(t_2)$ conditional on $G(t_1)$, then $Y(t_2) = X(G(t_2))$ conditional on $(G(t_1), G(t_2), Y(t_1))$, and so on. We can then compute any $S(t_j)$ directly from $Y(t_j)$.

It is convenient to view the sampling of $(G(t_j), Y(t_j))$ conditional on $(G(t_{j-1}), Y(t_{j-1}))$ as one step (step j) of the Markov chain. The state of the chain at step $j-1$ can be taken as $X_{j-1} = (G(t_{j-1}), Y(t_{j-1}), \bar{S}_{j-1})$, so we have a three-dimensional state, and we need two independent uniform random numbers at each step, one to generate $G(t_j)$ and the other to generate $Y(t_j) = X(G(t_j))$ given $(G(t_{j-1}), G(t_j), Y(t_{j-1}))$, both by inversion. Applying Array-RQMC with this setting would require a five-dimensional RQMC point set at each step, unless we can map the state to a lower-dimensional representation.

However, a key observation here is that the distribution of the increment $\Delta Y_j = Y(t_j) - Y(t_{j-1})$ depends only on the increment $\Delta_j = G(t_j) - G(t_{j-1})$ and not on $G(t_{j-1})$. This means that there is no need to memorize the latter in the state! Thus, we can define the state at step j as the two-dimensional vector $X_j = (Y(t_j), \bar{S}_j)$, or equivalently $X_j = (S(t_j), \bar{S}_j)$, and apply Array-RQMC with a four-dimensional RQMC point set if we use a two-dimensional sort for the states, and a three-dimensional RQMC point set if we map the states to a one-dimensional representation (using a Hilbert curve or a linear combination of the coordinates, for example). At step j, we generate $\Delta_j \sim \text{Gamma}((t_j - t_{j-1})/\nu, \nu)$ by inversion using a uniform random variate $U_{j,1}$, i.e., via $\Delta_j = F_j^{-1}(U_{j,1})$ where F_j is the cdf of the Gamma$((t_j - t_{j-1})/\nu, \nu)$

distribution, then ΔY_j by inversion from the normal distribution with mean $\theta\Delta_j$ and variance $\sigma^2\Delta_j$, using a uniform random variate $U_{j,2}$. Algorithm 3 summarizes this procedure. The symbol Φ denotes the standard normal cdf. We have

$$X_j = (Y(t_j),\bar{S}_j) = \varphi_j(Y(t_{j-1}), \bar{S}_{j-1}, U_{j,1}, U_{j,2})$$

where φ_j is defined by the algorithm. The payoff function is $g(X_c) = \bar{S}_c = \bar{S}$.

Algorithm 3 Computing $X_j = (Y(t_j),\bar{S}_j)$ given $(Y(t_{j-1}), \bar{S}_{j-1})$, for $1 \le j \le \tau$.

Generate $U_{j,1}, U_{j,2} \sim$ Uniform(0,1), independent;
$\Delta_j = F_j^{-1}(U_{j,1}) \sim$ Gamma$((t_j - t_{j-1})/\nu, \nu)$;
$Z_j = \Phi^{-1}(U_{j,2}) \sim$ Normal(0,1);
$Y(t_j) \leftarrow Y(t_{j-1}) + \theta\Delta_j + \sigma\sqrt{\Delta_j} Z_j$;
$S(t_j) \leftarrow S(0)\exp[(r+\omega)t_j + Y(t_j)]$;
$\bar{S}_j = [(j-1)\bar{S}_{j-1} + S(t_j)]/j$;

With this two-dimensional state representation, if we use a split sort or batch, we need four-dimensional RQMC points. With the Hilbert-curve sort, we only need three-dimensional RQMC points. We also tried a simple linear mapping $h_j : \mathbb{R}^2 \to \mathbb{R}$ defined by $h_j(S(t_j), \bar{S}_j) = b_j\bar{S}_j + (1 - b_j)S(t_j)$ where $b_j = (j-1)/(\tau - 1)$. At each step j, this h_j maps the state X_j to a real number $h_j(X_j)$, and we sort the states by increasing order of their value of $h_j(X_j)$. It uses a convex linear combination of $S(t_j)$ and $\bar{S}_j$ whose coefficients depend on j. The rationale for the (heuristic) choice of b_j is that in the late steps (when j is near τ), the current average $\bar{S}_j$ is more important (has more predictive power for the final payoff) than the current $S(t_j)$, whereas in the early steps, the opposite is true.

We made an experiment with the following model parameters, taken from Avramidis and L'Ecuyer (2006): $\theta = -0.1436$, $\sigma = 0.12136$, $\nu = 0.3$, $r = 0.1$, $T = 240/365$, $\tau = 10$, $t_j = 24j/365$ for $j = 1,\ldots,\tau$, $K = 100$, and $S(0) = 100$. The time unit is one year, the horizon is 240 days, and there is an observation time every 24 days. The exact value of the expected payoff for the Asian option is $\mu \approx 8.36$, and the MC variance per run is $\sigma_y^2 = \mathrm{Var}[Y_a] \approx 59.40$.

Table 5.1 summarizes the results. For each selected sorting method and point set, we report the estimated slope $\hat{\beta}$ for the linear regression model of $\log_2 \mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$ as a function of $\log_2(n)$ obtained from $m = 100$ independent replications with $n = 2^e$ for $e = 16,\ldots,20$, as well as the variance reduction factors (VRF) observed for $n = 2^{20}$ (about one million

Sort	Point sets	$\hat{\beta}$	VRF20
Split sort	MC	-1	1
	Stratif	-1.17	42
	Sobol'+LMS	-1.77	91550
	Sobol'+NUS	-1.80	106965
	Lattice+baker	-1.83	32812
Batch sort ($n_1 = n_2$)	MC	-1	1
	Stratif	-1	42
	Sobol'+LMS	-1.71	100104
	Sobol'+NUS	-1.54	90168
	Lattice+baker	-1.95	58737
Hilbert sort (with logistic map)	MC	-1	1
	Stratif	-1.43	204
	Sobol'+LMS	-1.59	68297
	Sobol'+NUS	-1.67	79869
	Lattice+baker	-1.55	45854
Linear map sort	MC	-1	1
	Stratif	-1.35	192
	Sobol'+LMS	-1.64	115216
	Sobol'+NUS	-1.75	166541
	Lattice+baker	-1.72	68739

Table 5.1. Regression slopes $\hat{\beta}$ for $\log_2 \mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$ vs $\log_2(n)$, and VRF compared with MC for $n = 2^{20}$, denoted VRF20, for the Asian option under the VG model

samples), denoted VRF20. For MC, the exact slope (or convergence rate) β is known to be $\beta = -1$. We see from the table that Array-RQMC provides much better convergence rates (at least empirically), and reduces the variance by very large factors for $n = 2^{20}$. Interestingly, the largest factors are obtained with the Sobol' points combined with our heuristic linear map sort, although the other sorts are also doing quite well. Figure 5.1 shows plots of $\log_2 \mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$ vs $\log_2(n)$ for selected sorts. It gives an idea of how well the linear model fits in each case.

There are other ways of defining the steps of the Markov chain for this example. For example, one can have one step for each Uniform(0,1) random number that is generated. This would double the number of steps, from c to $2c$. We generate Δ_1 in the first step, $Y(t_1)$ in the second step, Δ_2 in the third step, $Y(t_2)$ in the fourth step, and so on. Generating a single uniform per step instead of two reduces by 1 the dimension of the required RQMC point set. At odd step numbers, when we generate a Δ_j, the state can still be taken as $(Y(t_{j-1}), \bar{S}_{j-1})$ and we only need three-dimensional RQMC points, so we save one dimension.

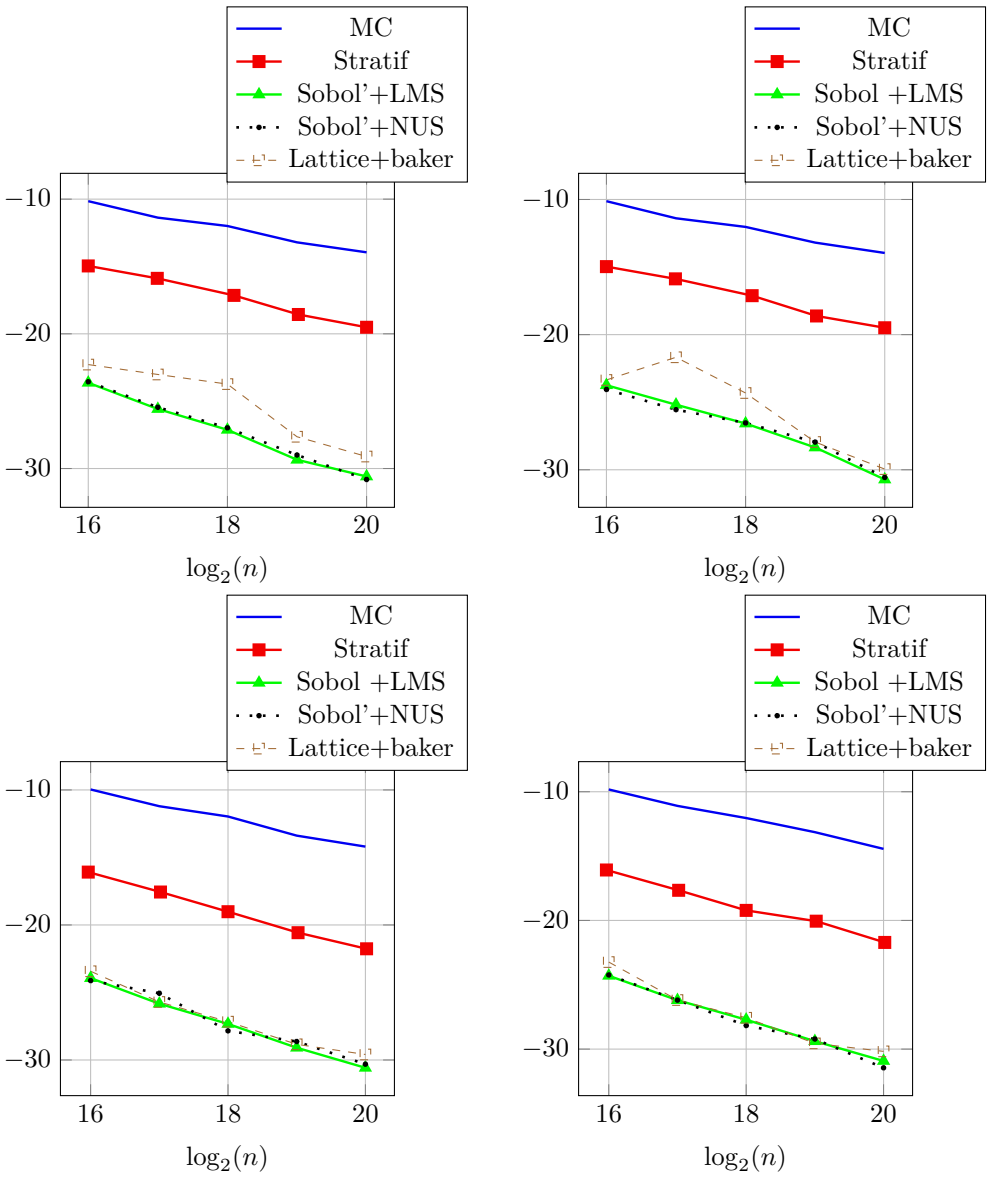

Figure 5.1. Plots of empirical $\log_2 \mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$ vs $\log_2(n)$ for various sorts and point sets, based on $m = 100$ independent replications. Above: with split sort (Left) and batch sort (right) and Below: with Hilbert sort (Left) and linear map sort (right).

But at even step numbers, we need Δ_j to generate $Y(t_j)$, so we need a three-dimensional state $(Y(t_{j-1}), \Delta_j, \bar{S}_{j-1})$ and four-dimensional RQMC points. We tried this approach and it did not perform better than the one described earlier, with two uniforms per step. It is also more complicated to implement.

Avramidis and L'Ecuyer (2006); Avramidis et al. (2003) describe other ways of simulating the VG process, for instance Brownian and gamma bridge sampling (BGBS) and difference of gammas bridge sampling (DGBS). BGBS generates first $G(t_c)$ then $Y(t_c)$, then conditional

on this it generates $G(t_{c/2})$ then $Y(t_{c/2})$ (assuming that c is even), and so on. DGBS writes the VG process Y as a difference of two independent gamma processes and simulate both using the bridge idea just described: first generate the values of the two gamma processes at t_c, then at $t_{c/2}$, etc. When using classical RQMC, these sampling methods brings an important variance reduction compared with the sequential one we use here for our Markov chain. Combining them with Array-RQMC is impractical, however, because the dimension of the state (the number of values that we need to remember and use for the sorting) grows up to about c, which is much to high, and the implementation is much more complicated. Also, these methods are effective when c is a power or 2 and $t_j = jT/c$, because then the conditional sampling for the gamma process is always from a symmetrical beta distribution and there is an efficient inversion method for that but they are less effective otherwise (L'Ecuyer and Simard, 2006). For comparison, we made an experiment using classical RQMC with these methods for the same numerical example as given here, but with $c = 8$ and $c = 16$ instead of $c = 10$, to have powers of 2, and $t_j = jT/c$. For Sobol'+LMS with $n = 2^{20}$, for $d = 16$, the values of VRF20 for BGSS, BGBS, and DGBS were 85, 895, 550, respectively. For $d = 8$, these values were 183, 1258, and 3405. The VRF20 values for Sobol'+LMS in Table 5.1 and are much larger that these, showing that Array-RQMC can provide much larger variance reductions.

For this VG model, we do not report results on the European option with Array-RQMC, because the Markov chain would have only one step: We can generate directly $G(t_c)$ and then $Y(t_c)$. For this, ordinary RQMC works well enough (L'Ecuyer, 2018).

5.5. Option Pricing Under The Heston Volatility Model

The Heston volatility model is defined by the following two-dimensional stochastic differential equation:

$$\begin{aligned} \mathrm{d}S(t) &= rS(t)\mathrm{d}t + V(t)^{1/2}S(t)\mathrm{d}B_1(t), \\ \mathrm{d}V(t) &= \lambda(\sigma^2 - V(t))\mathrm{d}t + \xi V(t)^{1/2}\mathrm{d}B_2(t), \end{aligned}$$

for $t \geq 0$, where (B_1,B_2) is a pair of standard Brownian motions with correlation ρ between them, r is the risk-free rate, σ^2 is the long-term average variance parameter, λ is the rate of return to the mean for the variance, and ξ is a volatility parameter for the variance.

The processes $S = \{S(t), t \geq 0\}$ and $V = \{V(t), t \geq 0\}$ represent the asset price and the volatility, respectively, as a function of time. We will examine how to estimate the price of European and Asian options with Array-RQMC under this model. Since we do not know how to generate $(S(t+\delta), V(t+\delta))$ exactly from its conditional distribution given $(S(t), V(t))$ in this case, we have to discretize the time. For this, we use the Euler method with τ time steps of length $\delta = T/\tau$ to generate a skeleton of the process at times $w_j = j\delta$ for $j = 1, \ldots, \tau$, over $[0,T]$. For the Asian option, we assume for simplicity that the observation times $t_1, \ldots, t_c$ used for the payoff are all multiples of δ, so each of them is equal to some w_j.

		European		Asian	
Sort	Point sets	$\hat{\beta}$	VRF20	$\hat{\beta}$	VRF20
Split sort	MC	-1	1	-1	1
	Stratif	-1.26	103	-1.29	38
	Sobol'+LMS	-1.59	44188	-1.48	6684
	Sobol'+NUS	-1.46	30616	-1.46	5755
	Lattice+baker	-1.50	26772	-1.55	5140
Batch sort	MC	-1	1	-1	1
	Stratif	-1.24	91	-1.25	33
	Sobol'+LMS	-1.66	22873	-1.23	815
	Sobol'+NUS	-1.72	30832	-1.38	1022
	Lattice+baker	-1.75	12562	-1.22	762
Hilbert sort (with logistic map)	MC	-1	1	-1	1
	Stratif	-1.26	43	-1.05	29
	Sobol'+LMS	-1.14	368	-0.87	39
	Sobol'+NUS	-1.06	277	-1.11	49
	Lattice+baker	-1.12	250	-0.89	42

Table 5.2. Regression slopes $\hat{\beta}$ for $\log_2 \mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$ vs $\log_2(n)$, and VRF compared with MC for $n = 2^{20}$, denoted VRF20, for the Asian option under the Heston model.

Following Giles (2008), to reduce the bias due to the discretization, we make the change of variable $W(t) = e^{\lambda t}(V(t) - \sigma^2)$, with $\mathrm{d}W(t) = e^{\lambda t}\xi V(t)^{1/2}\mathrm{d}B_2(t)$, and apply the Euler method to (S,W) instead of (S,V). The Euler approximation scheme with step size δ applied to W gives

$$\widetilde{W}(j\delta) = \widetilde{W}((j-1)\delta) + e^{\lambda(j-1)\delta}\xi(\widetilde{V}((j-1)\delta)\delta)^{1/2}Z_{j,2}.$$

Rewriting it in terms of V by using the reverse identity $V(t) = \sigma^2 + e^{-\lambda t}W(t)$, and after some manipulations, we obtain the following discrete-time stochastic recurrence, which we

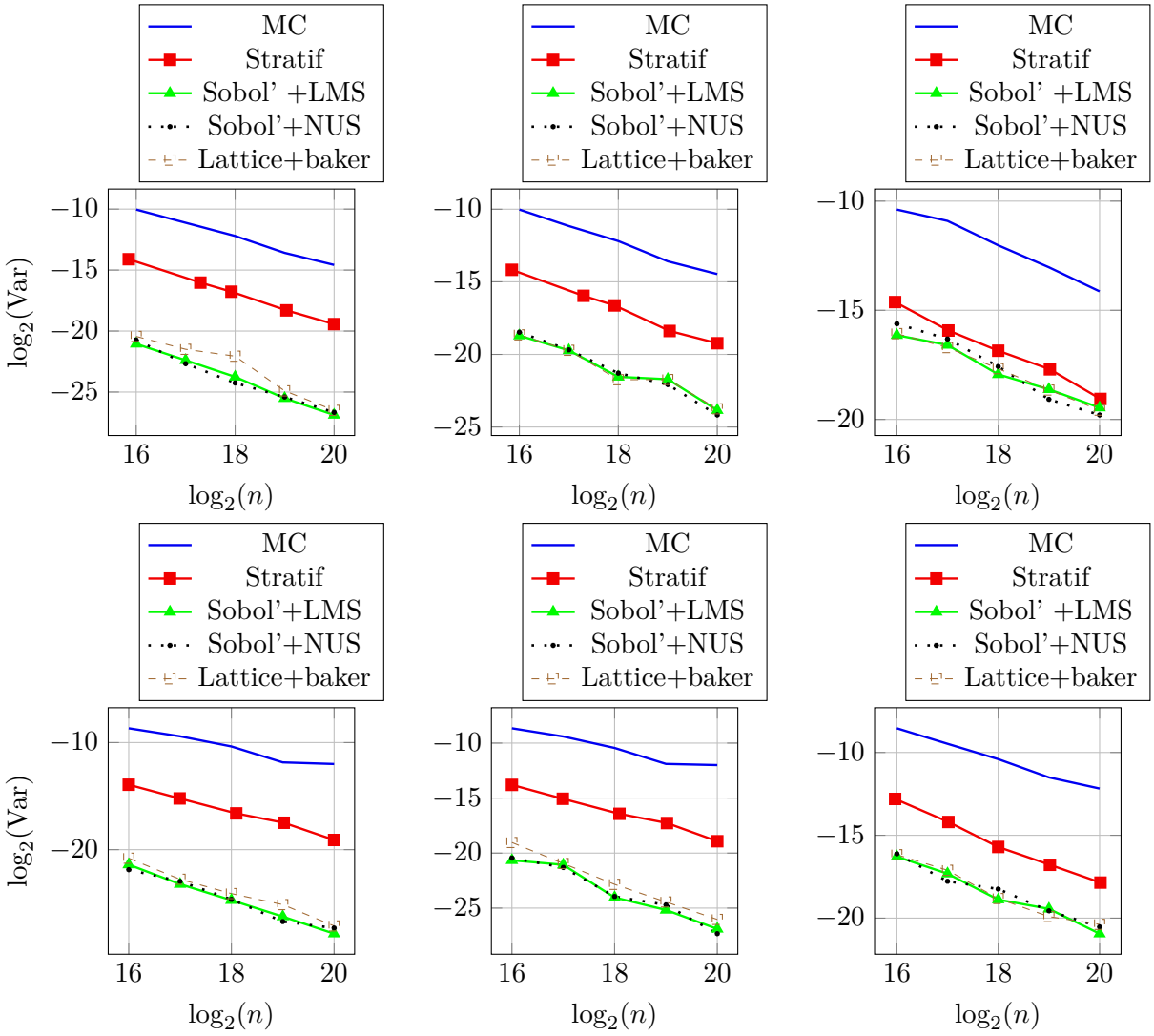

Figure 5.2. Plots of empirical $\log_2 \mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$ vs $\log_2(n)$ for various sorts and point sets, based on $m = 100$ independent replications, for the Heston model. Asian option (above) and European option (below), with split sort (left), batch sort (middle), and Hilbert sort (right).

will simulate by Array-RQMC:

$$\begin{aligned}\widetilde{V}(j\delta) &= \max\left[0, \sigma^2 + e^{-\lambda\delta}\left(\widetilde{V}((j-1)\delta) - \sigma^2 + \xi(\widetilde{V}((j-1)\delta)\delta)^{1/2} Z_{j,2}\right)\right], \\ \widetilde{S}(j\delta) &= (1 + r\delta)\widetilde{S}((j-1)\delta) + (\widetilde{V}((j-1)\delta)\delta)^{1/2}\widetilde{S}((j-1)\delta) Z_{j,1},\end{aligned}$$

where $(Z_{j,1}, Z_{j,2})$ is a pair of standard normals with correlation ρ. We generate this pair from a pair $(U_{j,1}, U_{j,2})$ of independent Uniform(0,1) variables via $Z_{j,1} = \Phi^{-1}(U_{j,1})$ and $Z_{j,2} = \rho Z_{j,1} + \sqrt{1-\rho^2}\,\Phi^{-1}(U_{j,2})$. We then approximate each $S(j\delta)$ by $\widetilde{S}(j\delta)$. The running average $\bar{S}_j$ at step j must be the average of the $S(t_k)$ at the observation times $t_k \le w_j = j\delta$. If

we denote $N_j = \sum_{k=1}^{c} \mathbb{I}[t_k \le j\delta]$, we have $\bar{S}_j = (1/N_j)\sum_{k=1}^{N_j} S(t_k)$, which we approximate by $\overline{\widetilde{S}}_j = (1/N_j)\sum_{k=1}^{N_j} \widetilde{S}(t_k)$. Here, the state of the chain is $X_j = (\widetilde{S}(j\delta), \widetilde{V}(j\delta))$ when pricing the European option and $X_j = (\widetilde{S}(j\delta), \widetilde{V}(j\delta), \overline{\widetilde{S}}_j)$ when pricing the Asian option. And two uniform random numbers, $(U_{j,1}, U_{j,2})$, are required at each step of the chain. We thus need four-dimensional RQMC point sets for the European option and five-dimensional RQMC point sets for the Asian option, if we do not map the state to a lower-dimensional representation. If we map the state to one dimension, as in the Hilbert curve sort, then we only need three-dimensional RQMC points for both option types.

We tried an alternative Markov chain definition in which the chain advances by one step each time a uniform random number is used, as in the VG example, to reduce the dimension of the RQMC points, but this gave no improvement.

We ran experiments with $T = 1$ (one year), $K = 100$, $S(0) = 100$, $V(0) = 0.04$, $r = 0.05$, $\sigma = 0.2$, $\lambda = 5$, $\xi = 0.25$, $\rho = -0.5$, and $c = \tau = 16$. This gives $\delta = 1/16$, so the time discretization for Euler is very coarse, but a smaller δ gives similar results in terms of variance reduction by Array-RQMC. For example, we ran experiments with $\tau = 256$ instead of $\tau = 16$ and the VRF20's had approximately the same sizes. Table 5.2 reports the estimated slopes $\hat{\beta}$ and VRF20, as in Table 5.1. Again, we observe large variance reductions and improved convergence rates from Array-RQMC. The best results are obtained with the split sort. Figure 5.2 shows plots of $\log_2 \mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$ vs $\log_2(n)$ for selected sorts.

5.6. Option Pricing Under The Ornstein-Uhlenbeck Volatility Model

The Ornstein-Uhlenbeck volatility model is defined by the following stochastic differential equations:

$$\begin{aligned} \mathrm{d}S(t) &= rS(t)\mathrm{d}t + e^{V(t)}S(t)\mathrm{d}B_1(t), \\ \mathrm{d}V(t) &= \alpha(b - V(t))\mathrm{d}t + \sigma\mathrm{d}B_2(t), \end{aligned}$$

for $t \ge 0$, where (B_1, B_2) is a pair of standard Brownian motions with correlation ρ between them, r is the risk-free rate, b is the long-term average volatility, α is the rate of return to the average volatility, and is σ a variance parameter for the volatility process. The processes $S = \{S(t), t \ge 0\}$ and $V = \{V(t), t \ge 0\}$ represent the asset price and the volatility process. We simulate these processes using Euler's method with τ time steps of length δ, as we did

for the Heston model, but without a change of variable. The discrete-time approximation of the stochastic recurrence is

$$\begin{aligned}\widetilde{S}(j\delta) &= \widetilde{S}((j-1)\delta) + r\delta\widetilde{S}((j-1)\delta) + \exp\left[\widetilde{V}((j-1)\delta)\right]\sqrt{\delta}Z_{j,1},\\ \widetilde{V}(j\delta) &= \alpha\delta b + (1-\alpha\delta)\widetilde{V}((j-1)\delta) + \sigma\sqrt{\delta}Z_{j,2},\end{aligned}$$

where $(Z_{j,1},Z_{j,2})$ is a pair of standard normals with correlation ρ. To generate this pair, we generate independent Uniform(0,1) variables $(U_{j,1}, U_{j,2})$, and put $Z_{j,1} = \Phi^{-1}(U_{j,1})$ and $Z_{j,2} = \rho Z_{j,1} + \sqrt{1-\rho^2}\,\Phi^{-1}(U_{j,2})$. For either the European or Asian option, the state of the Markov chain and the dimension of the RQMC points are the same as for the Heston model.

We ran a numerical experiment with $T = 1$, $K = 100$, $S(0) = 100$, $V(0) = 0.04$, $r = 0.05$, $b = 0.4$, $\alpha = 5$, $\sigma = 0.2$, $\rho = -0.5$, and $c = \tau = 16$ (so $\delta = 1/16$). Table 5.3 reports the estimated regression slopes $\hat{\beta}$ and VRF2. With $\tau = 256$ instead of $\tau = 16$, the VRF20's have about the same sizes.

		European		Asian	
Sort	Point sets	$\hat{\beta}$	VRF20	$\hat{\beta}$	VRF20
Batch sort	MC	-1	1	-1	1
	Stratif	-1.28	111	-1.23	29
	Sobol'+LMS	-1.35	61516	-1.22	4558
	Sobol'+NUS	-1.31	56235	-1.22	5789
	Lattice+baker	-1.37	61318	-1.20	5511
Hilbert sort (with logistic map)	MC	-1	1	-1	1
	Stratif	-1.40	440	-1.37	250
	Sobol'+LMS	-1.52	194895	-1.40	41100
	Sobol'+NUS	-1.68	191516	-1.37	39861
	Lattice+baker	-1.59	165351	-1.47	37185

Table 5.3. Regression slopes $\hat{\beta}$ for $\log_2 \mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$ vs $\log_2(n)$, and VRF compared with MC for $n = 2^{20}$, denoted VRF20, for the European and Asian options under the Ornstein-Uhlenbeck model.

Conclusion

We have shown how Array-RQMC can be applied for pricing options under stochastic volatility models, and gave detailed examples with the VG, Heston, and Ornstein-Uhlenbeck models. With the models, the method requires higher-dimensional RQMC points than with the simpler GBM model studied previously, and when time has to be discretized to apply

Euler's method, the number of steps of the Markov chain is much larger. For these reasons, it was not clear a priori if Array-RQMC would be effective. Our empirical results show that it brings very significant variance reductions compared with crude Monte Carlo.

Acknowledgments

This work has been supported by a discovery grant from NSERC-Canada, a Canada Research Chair, and a Grant from the IVADO Fundamental Research Program, to P. L'Ecuyer.

Chapter 6

Article 4: Variance Reduction with Array-RQMC for Tau-Leaping Simulation of Stochastic Biological and Chemical Reaction Networks

In the fourth article, we consider systems of chemical species whose molecule numbers dynamically change over time as the molecules react via a set of predefined chemical equations. The evolution of such systems is typically modeled by a continuous-time Markov chain (CTMC) whose state is a vector that gives the number of copies of each species. The τ -leaping method is designed to model these systems by approximating the CTMC by a discrete-time Markov chain (DTMC), which can be simulated by generating a vector of independent Poisson random variables at each step, and updating the state to reflect all the reactions that occurred during a fixed time interval. We investigate the use of Array-RQMC, to reduce the variance when simulating such systems with τ-leaping. When applying the method properly and introducing new sorting methods, we find that variance reductions can be achieved by factors in the thousands. These factors are far greater than those previously found by other authors who have tried RQMC methods on the same examples.

This article has been submitted for publication in The Bulletin of Mathematical Biology. Preliminary work was presented at the following conferences:

— The Optimization Days, Montreal, May 2019;

— 12-th International Conference on Monte Carlo Methods and Applications, Sydney, Australia, July 2019.

The main author contributions are:

— The general research ideas were proposed by Pierre L'Ecuyer;

— The research (including implementation, experiments, etc.) was carried out jointly by Amal Ben Abdellah and Florian Puchhammer;

— The article was written jointly by all three authors.

Variance Reduction with Array-RQMC for Tau-Leaping Simulation of Stochastic Biological and Chemical Reaction Networks

Florian Puchhammer[1], Amal Ben Abdellah[2] and Pierre L'Ecuyer[2]

[1] Basque Center for Applied Mathematics, Alameda de Mazarredo 14, 48009 Bilbao, Basque Country, Spain; and DIRO, Université de Montréal, C.P. 6128, Succ. Centre-Ville, Montréal, H3C 3J7, Canada.

[2] Département d'Informatique et de Recherche Opérationnelle, Pavillon Aisenstadt, Université de Montréal, C.P. 6128, Succ. Centre-Ville, Montréal, Québec, Canada H3C 3J7.

Abstract

We explore the use of Array-RQMC, a randomized quasi-Monte Carlo method designed for the simulation of Markov chains, to reduce the variance when simulating stochastic biological or chemical reaction networks with τ-leaping. We find that when the method is properly applied, variance reductions by factors in the thousands can be obtained. These factors are much larger than those observed previously by other authors who tried RQMC methods for the same examples. Array-RQMC simulates an array of realizations of the Markov chain and requires a sorting function to reorder these chains according to their states, after each step. The choice of a good sorting function is a key ingredient for the efficiency of the method. We illustrate this by comparing various choices. The expected number of reactions of each type per step also has an impact on the efficiency gain.

Key words: Chemical reaction networks, stochastic biological systems, variance reduction, Quasi-Monte Carlo, Array-RQMC, Tau-leaping, continuous-time Markov chains, Gillespie.

6.1. Introduction

We consider systems of chemical species whose molecule numbers dynamically change over time as the molecules react via a set of predefined chemical equations. The evolution of such systems is typically modeled by a *continuous-time Markov chain* (CTMC) (Anderson and Kurtz, 2011; Anderson, 1991; Gillespie, 1977) whose state is a vector that gives the

number of copies of each species. Each transition (or jump) of the CTMC corresponds to the occurrence of one reaction, and the occurrence rate of each potential reaction (also called the *reaction propensity*) is a function of the state of the chain. The probability that any given reaction is the next in order is proportional to its propensity and the time until the next reaction has an exponential distribution whose rate is the sum of these propensities. The *stochastic simulation algorithm* (SSA) of Gillespie (1977) simulates the successive transitions of this CTMC one by one, by generating the exponential time until the next reaction and determining independently which reaction it is. This method is exact (there is no bias). But when the number of molecules is large, simulating all the reactions one by one is often too slow, because their frequency is too high. One popular alternative is to approximate the CTMC by a *discrete-time Markov chain* (DTMC), as follows. Fix a time interval $\tau > 0$. Under the simplifying assumption that the rates of the different reactions do not change during the next τ units of time, the numbers of occurrences for each type of reaction are independent Poisson random variables with means that are τ times the occurrence rates (or propensities) of these reactions. Each step (or transition) of the DTMC corresponds to τ units of time for the CTMC. This DTMC can be simulated by generating a vector of independent Poisson random variables at each step, and updating the state to reflect all the reactions that occurred during this time interval. In the setting of chemical reaction networks, this approach is the *τ-leaping method* of Gillespie (2001), and it is widely used in practice. This is the method we consider in this paper.

There are several other approximation methods, some of them leading to simpler and faster simulations, but the error and/or bias can also be more significant (Gillespie, 2000; Higham, 2008). One simple approach uses a fluid approximation in which the copy numbers are assumed to take real values that vary in time according to a system of deterministic differential equations called the *reaction rate equations* which can be simulated numerically (Gillespie, 2000; Higham, 2008). This type of deterministic model is the primary tool in the field of system dynamics, and it is widely used in many areas. It corresponds to chemical kinetics equations found in textbooks. But this model ignores randomness, so it cannot capture the stochastic variations observed in experiments with real systems (Beentjes and Baker, 2019). Noise can be introduced via a *stochastic differential equations* model, which amounts

essentially to approximate the Poisson distribution by a normal one, and the denumerable-state CTMC by a continuous-state process. This leads to the *chemical Langevin equation* (Beentjes and Baker, 2019; Gillespie, 2000), which can be simulated efficiently by standard methods for stochastic differential equations (Kloeden and Platen, 1992) and may provide a reasonable approximation when the number of molecules of each type is very large, but can otherwise suffer from bias.

The purpose of the stochastic simulations with τ-leaping could be for example changing to estimate the probability distribution of the state at a given time t, or the probability that the state is in a given subset at time t, or perhaps the expectation of some function of the state. The simulations are usually done via Monte Carlo (MC) sampling, using a random number generator that provides a good imitation of independent uniform random variables over the interval (0,1) (L'Ecuyer, 2012). For MC estimators based on the average over n independent samples, the variance and standard deviation converge as $\mathcal{O}(n^{-1})$ and $\mathcal{O}(n^{-1/2})$, respectively, which is rather slow.

Randomized quasi-Monte Carlo (RQMC) provides an alternative sampling approach which under favorable conditions can improve this convergence rate of the variance to $\mathcal{O}(n^{-2+\epsilon})$ for any $\epsilon > 0$, and even better in special situations (L'Ecuyer, 2009, 2018; L'Ecuyer and Lemieux, 2002; Owen, 1997, 2003). *Quasi-Monte Carlo* (QMC) replaces the n independent vectors of uniform random numbers that drive the simulations by n *deterministic* vectors with a sufficient number of coordinates to simulate the system and which cover the space (the unit hypercube) more evenly than typical independent random points (Dick and Pillichshammer, 2010; Niederreiter, 1992). RQMC randomizes these points in a way that each individual point becomes a vector of independent uniform random numbers, while at the same time the set of points as a whole retains its structure and high uniformity. As a result, RQMC can provide an unbiased estimator with lower variance.

On the other hand, there are two important limitations. Firstly, the $\mathcal{O}(n^{-2+\epsilon})$ convergence rates for RQMC are proved only under conditions that the integrand is a smooth function of the uniforms, whereas when simulating the CTMC considered here, the sequence of states that are visited is discontinuous in the underlying uniform random variates. Secondly, when the points are high-dimensional and some high-order interaction between the coordinates are important, the variance reduction is usually limited, and this often happens

when simulating the CTMCs that model reaction networks via either direct SSA or τ-leaping. Indeed, those simulations require at least one or two random numbers per step of the chain, the number of steps can be very large in real applications, so the dimension of the points, which is the total number of random numbers that are required to simulate one realization of the process, can be very large. Beentjes and Baker (2019) investigated the performance of τ-leaping combined with traditional RQMC and found that the gain from RQMC compared to MC was small. They mentioned the two limitations above as possible explanations for this behavior.

The *Array-RQMC* algorithm (L'Ecuyer et al., 2006, 2008, 2009) has been developed precisely to recapture the power of RQMC when simulating Markov chains over a large number of steps, as in the problem considered here. The empirical variance under array-RQMC has been observed to converge faster than under MC in several examples from various areas, sometimes at the $n^{-2+\epsilon}$ rate, even for some examples where the cost function was discontinuous (Ben Abdellah et al., 2019b; Demers et al., 2005; Dion and L'Ecuyer, 2010; L'Ecuyer et al., 2007, 2008, 2009, 2018). The faster convergence has also been proven theoretically under certain conditions (L'Ecuyer et al., 2008).

Our present work was motivated by Beentjes and Baker (2019) and our aim was to see how Array-RQMC can improve upon MC and classical RQMC, first by using the same examples as in their paper. Hellander (2008) also experimented with Array-RQMC, in combination with uniformization of the CTMC and conditional Monte Carlo (CMC) based on the discrete-time conversion method of Fox and Glynn (1990). Their goal was to estimate the probability distribution of the state at a fixed time $t > 0$. In this setting, CMC alone provably reduces the variance. Empirically, with CMC, they obtained variance reductions by factors of about 20 in one example and 45 in another example. With the combination of CMC with Array-RQMC, they observed variance reductions by a factor of about 100 with $n = 10^5$ for both examples. Thus, Array-RQMC provides an additional gain on top of CMC, by a factor of about 2.5 to 5.

In this paper, we show how to obtain much larger variance-reduction factors with Array-RQMC. We do this in the same setting as Beentjes and Baker (2019), where the τ-leaping method is used to estimate an expectation at a given time t. We find empirically that the

combination of τ-leaping with the Array-RQMC algorithm can bring not only a significant variance reduction, but also an improved convergence rate, compared with plain MC.

The main idea of the Array-RQMC algorithm is to simulate n copies of the Markov chain in parallel, in a way that the empirical distribution of the chain's states at any given step is closer to the exact theoretical distribution at that step than with ordinary MC. To achieve this, at each step, the first few coordinates of the RQMC point set are designated to match the points to the states, and the remaining coordinates are used to advance the chains by one step. This matching can be interpreted as sorting the chains in some particular order, to match the ordering of the RQMC points. In the simple case where the state is one-dimensional, it suffices to enumerate the points by increasing order of their first coordinate and sort the chains by increasing order of their state. For higher-dimensional states, one possibility is to use some kind of multivariate sort to order both the points and the states; we will describe several of these sorts in Section 6.3.2. Another approach is to define an *importance function*, which maps the state to a one-dimensional representative value, and sort the chains by that value. The choice of mapping can have a significant impact on the performance. If the mapping is fast to evaluate, this approach can reduce the computing time significantly, because a one-dimensional sort is usually much faster to execute than a multivariate one. To preserve the power of Array-RQMC, on the other hand, the importance function must provide a good estimate (or forecast) of the expected future value or cost, given the state at which it is evaluated. For this, it must be tailored to the problem at hand. A good tradeoff between simplicity and prediction accuracy is not always easy to achieve, but it is a key ingredient for the performance of Array-RQMC. As a proof of concept that this approachcan work for reaction networks, we experiment with a very simple one-step look-ahead importance function, and we find that it works very well in all our examples. Empirically, in our experiments, this approach is often competitive with the best multivariate sorts interms of variance reduction, and the sorting times are shorter, so it often provides the best efficiency improvement. We also discuss how more elaborate importance functions could be defined.

The remainder is structured as follows. In Section 6.2 we recall the fixed step τ-leaping method for the simulation of well-mixed reaction networks in its simplest form. In Section 6.3, we define the Array-RQMC method and discuss some of the most prominent multivariate

sorting algorithms. In Section 6.4, we describe the methodology used for our experiments and provide numerical results, with a discussion. A conclusion follows.

6.2. The CTMC Model and the τ-Leaping Algorithm for Reaction Networks

We consider a system comprised of $\ell \geq 1$ types of chemical species $S_1, \ldots, S_\ell$ that can react via $d \geq 1$ reaction types (or channels) $R_1, \ldots, R_d$. We assume that the species are well-mixed within a volume that does not change over time and whose temperature remains constant. Each reaction R_k, $k = 1, \ldots, d$, can be written as

$$\alpha_{1,k} S_1 + \cdots + \alpha_{\ell,k} S_\ell \xrightarrow{c_k} \beta_{1,k} S_1 + \cdots + \beta_{\ell,k} S_\ell, \quad \alpha_{i,k}, \beta_{i,k} \in \mathbb{N}_0,$$

where $c_k > 0$ is the reaction rate constant for R_k. Let $\mathbf{X}(t) = (X_1(t), ..., X_\ell(t)) \in \mathbb{N}_0^\ell$, where $X_i(t)$ is the copy number (i.e., the number of molecules) of type S_i at time t, for $i = 1, \ldots, \ell$ and $0 \leq t \leq T$. The process $\{\mathbf{X}(t), t \geq 0\}$ is modeled as a CTMC with fixed initial state $\mathbf{X}(0) = \mathbf{x}_0$ and for which each jump corresponds to the occurrence of one reaction. The jump rate (or *propensity function*) for reaction R_k is a function a_k of the current state; it is $a_k(\mathbf{x})$ when $\mathbf{X}(t) = \mathbf{x}$. This means that for a small $\delta > 0$, reaction R_k occurs exactly once during the time interval $(t, t + \delta]$ with probability $a_k(\mathbf{x})\delta + o(\delta)$ and occurs more than once with probability $o(\delta)$. When R_k occurs, the state changes from $\mathbf{x}$ to $\mathbf{x} + \boldsymbol{\zeta}_k$, where $\boldsymbol{\zeta}_k = (\beta_{1,k} - \alpha_{1,k}, \ldots, \beta_{\ell,k} - \alpha_{\ell,k})$ is the stoichiometric vector for R_k. The standard for $a_k(\mathbf{x})$, which we assume in our examples, is $a_k(\mathbf{x}) = c_k \mathcal{H}_k(\mathbf{x})$ where $\mathcal{H}_k(\mathbf{x}) = \prod_{i=1}^{\ell} \binom{x_i}{\alpha_{i,k}}$ represents the number of ways of selecting the molecules for reaction R_k when in state $\mathbf{x} = (x_1, \ldots, x_\ell)$ (Higham, 2008). When in state $\mathbf{x}$, the time until the next reaction has an exponential distribution with rate $\lambda(\mathbf{x}) = \sum_{k=1}^{d} a_k(\mathbf{x})$, the probability that this reaction is R_k is $a_k(\mathbf{x})/\lambda(\mathbf{x})$, and these random variables are independent. The SSA of Gillespie (1977) simulates this CTMC directly. However, when a very large number of reactions occur in the time interval of interest, the direct simulation approach may be too slow.

Gillespie (2001) proposed the *τ-leaping* algorithm as a way to speed up the simulation. This approach discretizes the time into intervals of length $\tau > 0$, and it generates directy the number of occurrences of each type of reaction in each such interval. If $\mathbf{X}(t) = \mathbf{x}$ at the beginning of an interval, it is assumed (as an approximation) that the rate of each reaction R_k

remains equal to $a_k(\mathbf{x})$ during the entire interval $[t, t+\tau]$. Under this simplifying assumption, the number D_k of occurrences of R_k during this time interval has a Poisson distribution with mean $a_k(\mathbf{x})\tau$, and $D_1, \ldots, D_d$ are independent. These D_k can be simulated easily via the inversion method, by generating independent uniform random numbers over (0,1) and applying the inverse of the cumulative distribution function (cdf) of the appropriate Poisson distribution (Giles, 2016). The simulated state at time $t+\tau$ is then $\mathbf{x}+\sum_{k=1}^{d} D_k \boldsymbol{\zeta}_k$. Repeating this at each step gives an approximating *discrete-time Markov chain* (DTMC) $\{\mathbf{X}_j, j \geq 0\}$ defined by $\mathbf{X}_0 = \mathbf{x}_0$ and

$$\mathbf{X}_j \;=\; \mathbf{X}_{j-1} + \sum_{k=1}^{d} D_{j,k}\boldsymbol{\zeta}_k, \;=\; \mathbf{X}_{j-1} + \sum_{k=1}^{d} F_{j,k}^{-1}(U_{j,k})\boldsymbol{\zeta}_k \;\stackrel{\text{def}}{=}\; \varphi(\mathbf{X}_{j-1}, \mathbf{U}_j), \tag{6.2.1}$$

where $D_{j,k} = F_{j,k}^{-1}(U_{j,k})$, $F_{j,k}$ is the cdf of the Poisson distribution with mean $a_k(\mathbf{X}_{j-1})\tau$, $\mathbf{U}_j = (U_{j,1}, \ldots, U_{j,d})$, and the $U_{j,k}$ are independent uniform random numbers over (0,1), for $k = 1, \ldots, d$ and $j \geq 1$. If τ is small enough, $\mathbf{X}_j$ has approximately the same distribution as $\mathbf{X}(j\tau)$, so this DTMC provides an approximate skeleton of a CTMC sample path.

This τ-leaping approximation has some potential problems, because it introduces bias which can propagate across successive steps, and this bias can be important if τ is not small enough. It is also possible to obtain negative copy numbers, i.e., some coordinates of some $\mathbf{X}_j$ taking negative values. Adaptive strategies and modifications of the algorithm have been designed to prevent or handle this; see, e.g., (Anderson and Higham, 2012; Anderson, 2008; Beentjes and Baker, 2019), and the references given there. We do not discuss these techniques in this paper. Our main goal is to explore how Array-RQMC can be effectively combined with τ-leaping, and we keep the setting simple to avoid distractions. In our experiments, we took τ small enough so we did not observe negative copy numbers.

Following Beentjes and Baker (2019), we suppose that the objective is to estimate $\mu = \mathbb{E}[g(\mathbf{X}(T))]$ for a given time $T > 0$ and some function $g : \mathbb{N}_0^\ell \to \mathbb{R}$. These authors only took a coordinate projection for g (i.e., they only estimated expected copy numbers) in their examples, and we do the same, but what we do applies easily to other choices of g.

For example, $g(\mathbf{x})$ could be the indicator that $\mathbf{x}$ belongs to a given set A, in which case $\mu = \mathbb{P}[\mathbf{X}(T) \in A]$. We take $\tau = T/s$ where s is a positive integer that represents the number of steps of the DTMC that will be simulated. To estimate μ with τ-leaping and MC, we

simulate n independent realizations of the DTMC via

$$\mathbf{X}_{i,0} = \mathbf{x}_0, \qquad \mathbf{X}_{i,j} = \varphi_j(\mathbf{X}_{i,j-1}, \mathbf{U}_{i,j}) \quad \text{for } j = 1, \ldots, s \text{ and } i = 0, \ldots, n-1, \tag{6.2.2}$$

where the $\mathbf{U}_{i,j}$'s are independent uniform random points over $(0,1)^d$. The estimator is

$$\hat{\mu}_n = \frac{1}{n} \sum_{i=0}^{n-1} g(\mathbf{X}_{i,s}). \tag{6.2.3}$$

We know that $\mathbb{E}[\hat{\mu}_n] = \mathbb{E}[g(\mathbf{X}_s)] \approx \mathbb{E}[g(\mathbf{X}(T))] = \mu$ (we do not look at the bias $\mathbb{E}[g(\mathbf{X}_s)] - \mathbb{E}[g(\mathbf{X}(T))]$ in this paper) and $\mathrm{Var}[\hat{\mu}_n] = \mathrm{Var}[g(\mathbf{X}_s)]/n$.

To use classical RQMC instead of MC, we simply replace the independent random points by a set of n vectors $\mathbf{V}_i = (\mathbf{U}_{i,1}, \mathbf{U}_{i,2}, \ldots, \mathbf{U}_{i,s})$, $i = 1, \ldots, n$, which form an RQMC point set in sd dimensions, as did Beentjes and Baker (2019).

6.3. Array-RQMC to Simulate the DTMC

6.3.1. The Array-RQMC Algorithm

We now explain how to apply Array-RQMC to simulate the DTMC via (6.2.2) and estimate $\mathbb{E}[g(\mathbf{X}_s)] \approx \mu$ again with (6.2.3), but with a different sampling strategy for the random numbers. The algorithm simulates the n sample paths of the DTMC in parallel, using an $(l + d)$-dimensional

RQMC point set to advance all the chains by one step at a time, for some $l \in \{1, \ldots, \ell\}$. The first l coordinates of the points are used to make a one-to-one pairing between the chains and the points, and the other d coordinates are used to advance the chains. When $l < \ell$, one must first define a *dimension-reduction mapping* $h : \mathbb{N}_0^\ell \to \mathbb{R}^l$ whose aim is to extract the most important features from the state and summarize them in a lower-dimensional vector which is used for the sort. For $l = 1$, the mapping h has been called an *importance function* or *sorting function* (L'Ecuyer et al., 2006, 2007, Section 3). At each step, both the RQMC points and the chains are ordered using the same l-dimensional sort. Different types of sorts are discussed in Section 6.3.2.

Specifically, we select a deterministic low-discrepancy (QMC) point set of the form $\tilde{\mathbf{Q}}_n = \{(\mathbf{w}_i, \mathbf{u}_i), i = 0, \ldots, n-1\}$, with $\mathbf{w}_i \in [0,1)^l$ and $\mathbf{u}_i \in [0,1)^d$, whose points are already sorted with respect to their first l coordinates with the multivariate sort that we have selected. At

each step j, we randomize the last d coordinates of the points of $\tilde{\mathbf{Q}}_n$ to obtain the RQMC point set

$$\mathbf{Q}_{n,j} = \{(\mathbf{w}_i, \mathbf{U}_{i,j}) : i = 0, \ldots, n-1\}, \tag{6.3.1}$$

in which each $\mathbf{U}_{i,j}$ is uniformly distributed in $[0,1)^d$. We also sort the n states $\mathbf{X}_{0,j-1}, \ldots, \mathbf{X}_{n-1,j-1}$ based on their values of $h(\mathbf{X}_{0,j-1}), \ldots, h(\mathbf{X}_{n-1,j-1})$, using the same sorting algorithm as for the QMC points, and let π_j denote the permutation of the indices $\{0,1,\ldots,n-1\}$ implicitly defined by this reordering. Then the n chains advance to step j via

$$\mathbf{X}_{i,j} = \varphi(\mathbf{X}_{\pi_j(i),j-1}, \mathbf{U}_{i,j}), \qquad i = 0, \ldots, n-1.$$

It is also possible to use a different sorting method at each step j, in which case the QMC points must be sorted differently as well, so this is usually not convenient.

At the end, one computes $\hat{\mu}_n$ in (6.2.3), which is an unbiased estimator of $\mathbb{E}[g(\mathbf{X}_s)]$. The main goal of this procedure is for the empirical distribution of the states $\mathbf{X}_{0,j}, \ldots, \mathbf{X}_{n-1,j}$ to better approximate the theoretical distribution of $\mathbf{X}_j$ at each step j, than if the chains were simulated independently with standard MC, and as a result reduce the variance of $\hat{\mu}_n$. For further theoretical analysis and empirical evidence, see for example L'Ecuyer et al. (2008, 2009, 2018). To estimate the variance of this Array-RQMC estimator, one can repeat the entire procedure m times, with independent randomizations of the points, and take the empirical variance of the m realizations of $\hat{\mu}_n$ as an unbiased estimator for $\mathrm{Var}[\hat{\mu}_n]$. This Array-RQMC procedure is stated in Algorithm 4.

Algorithm 4 Array-RQMC Algorithm

```
X_{i,0} ← x_0 for i = 0,...,n − 1;
for j = 1,2,...,s do
    Sort the states X_{0,j−1}, ..., X_{n−1,j−1} by their values of h(X_{i,j−1}),
        using the selected sort, and let π_j be the corresponding permutation;
    Randomize afresh the last d coordinates of the RQMC points, U_{0,j},...,U_{n−1,j};
    for i = 0,1,...,n − 1 do
        X_{i,j} = φ(X_{π_j(i),j−1}, U_{i,j}) ;
    end for
end for
Return the estimator μ̂_n = (1/n) Σ_{i=0}^{n−1} g(X_{i,s}).
```

6.3.2. Sorting Strategies

In the special case where $l = 1$, the RQMC points are sorted by their first coordinate and the states $\mathbf{X}_{i,j-1}$ are simply sorted by their value of $h(\mathbf{X}_{i,j-1})$, in increasing order. In this case, one would typically have $\mathbf{w}_i = i/n$ and the points are already sorted by construction (this is true for all the point sets used in this paper).

When $\ell > 1$, sorting for good pairing is less obvious. Two related multivariate sorts that gave good results for other applications are the batch sort and the split sort (El Haddad et al., 2008; Lécot and Coulibaly, 1998; L'Ecuyer et al., 2009, 2018). For the *batch sort* we factor $n = n_1 n_2 \cdots n_L$ with $L \geq 1$. Each time we sort, we split the set of states into n_1 batches of size n/n_1 such that the first coordinate of every state in one batch is smaller or equal to the first coordinate of every state in the next batch; then we further subdivide each batch into n_2 batches of size $n/(n_1 n_2)$ in the same way but now according to the second coordinate of the states. This procedure is repeated L times in total. If $L > \ell$, after ℓ steps we begin subdividing the batches with respect to their first coordinate again. The *split sort* is simply a variant of the batch sort in which $n = 2^L$ and $n_1 = n_2 = \ldots = n_L = 2$.

Another way of sorting is to map the states to the ℓ-dimensional unit hypercube $[0,1)^\ell$, so we can assume that the state space is now $[0,1)^\ell$ instead of $\mathbb{N}_0^\ell$, and then use a discretized version of a space filling curve for this hypercube. The hypercube is partitioned into a grid of small subcubes so that the event that two states fall in the same small subcube has a very small probability, then the states are sorted in the order that their subcubes are visited by the curve (those in the same subcube can be ordered arbitrarily). With this, we use $(d+1)$-dimensional RQMC points sorted by their first coordinate. This approach is in fact an implicit way to map the states to the one-dimensional real line, and then use a one-dimensional sort (with $l = 1$). This has been suggested in particular with a Z-curve (Wächter and Keller, 2008) and with a Hilbert curve (Gerber and Chopin, 2015). We call the latter a *Hilbert curve sort*. To map ℓ-dimensional states to $[0,1)^\ell$, Gerber and Chopin (2015) suggest applying a rescaled logistic transformation $\Psi(x_j) = 1/1/(1+\exp[-(x_j-\mu_j+2\sigma_j)/(4\sigma_j)])$, $1 \leq j \leq \ell$, to each coordinate. We estimated the means μ_j and the variances σ_j of the copy numbers of each species at every step, from data obtained from preliminary experiments.

A variant that avoids the need for such a transformation is the *Hilbert batch sort* (L'Ecuyer et al., 2018): One first applies a batch sort to partition $\mathbb{R}^\ell$ into n boxes, each of which

containing exactly one of the states, then these boxes are associated with n subcubes in $[0,1)^\ell$ and the states are enumerated in the order that the corresponding boxes are visited by the Hilbert curve.

All these multivariate sorts can be computationally expensive when n is large. For this reason, we made significant efforts in this work to explore ways of defining importance functions $h : \mathbb{N}_0^\ell \to \mathbb{R}$ that can be computed quickly during the simulations and provide at the same time good representations for the value of a state. An appropriate choice of h is certainly problem-dependent and good ones have been constructed for some examples in other settings such as computational finance, queueing, and reliability (Ben Abdellah et al., 2019b; L'Ecuyer et al., 2007, 2008, 2018).

We adopt the (partly heuristic) idea that at each step j, an ideal importance function h_j should have the property that $h_j(\mathbf{x})$ is a good approximation of $\mathbb{E}[g(\mathbf{X}_s) \mid \mathbf{X}_j = \mathbf{x}]$ for all $\mathbf{x} \in \mathbb{N}_0^\ell$ and $j = 1, \ldots, s$ (L'Ecuyer et al., 2007, 2009). To really do this, we need to construct a different approximation h_j for each j. We will call it a *step-dependent importance function* (SDIF). To see how well this general type of approach could perform, we made the following experiment with each of the examples considered in Section 6.4. First, we generated data by simulating the DTMC for $n = 2^{19}$ independent "pilot" samples paths, and we collected the n pairs $(\mathbf{X}_{i,j}, g(\mathbf{X}_{i,s}))$, $i = 0, \ldots, n-1$, for each j. Then, our aim was to find a function $h_j : \mathbb{N}_0^\ell \to \mathbb{R}$ for which $h_j(\mathbf{X}_{i,j})$ was a good predictor of $g(\mathbf{X}_{i,s})$. For this, we selected a parameterized form of function h_j, say $h_j(\boldsymbol{\theta},\cdot)$, which depends on a parameter vector $\boldsymbol{\theta}$, and we estimated the best value of $\boldsymbol{\theta}$ by least-squares regression from the data. The general form that we explored for $h_j(\boldsymbol{\theta},\mathbf{x})$ was a linear combination of polynomials in the coordinates of $\mathbf{x}$, where $\boldsymbol{\theta}$ was the vector of coefficients in the linear combination. The motivation for this choice is that the expected number of molecules of a given type at the next step, given the current state, is an affine function of the expected number of reactions of each type that will occur at that step, and this expected number for reaction type R_k is in turn linear in $a_k(\mathbf{x})$, which is a known polynomial in the coordinates of $\mathbf{x}$.

A cruder but less expensive strategy uses the same function $h_j = h$ for all j. One special case of this is to use h_{s-1} at all steps. We had some success with this simple version, which we call the *one-step look-ahead importance function* (OSLAIF). In the special case where $g(\mathbf{x})$ is linear in $\mathbf{x}$, say $g(\mathbf{x}) = \mathbf{b}^\mathsf{t}\mathbf{x}$, then $h_{s-1}(\mathbf{x}) \stackrel{\text{def}}{=} \mathbb{E}[g(\mathbf{X}_s) \mid \mathbf{X}_{s-1} = \mathbf{x}] = \mathbb{E}[\mathbf{b}^\mathsf{t}\mathbf{X}_1 \mid \mathbf{X}_0 = \mathbf{x}]$

is given by a polynomial in $\mathbf{x}$, and one can calculate this polynomial exactly, since

$$\mathbb{E}[\mathbf{X}_1 \mid \mathbf{X}_0 = \mathbf{x}] \;=\; \mathbf{x} + \sum_{k=1}^{d} \boldsymbol{\zeta}_k \mathbb{E}[D_{1,k} \mid \mathbf{X}_0 = \mathbf{x}] \;=\; \mathbf{x} + \tau \sum_{k=1}^{d} \boldsymbol{\zeta}_k a_k(\mathbf{x}), \tag{6.3.2}$$

which is a vector of polynomials in $\mathbf{x}$ that are easy to calculate. This includes the case of $g(\mathbf{x}) = x_i$, the number of molecules of species i, which occurs in all our examples.

Extending this to more than one step can be more difficult when the a_k are nonlinear. One can write

$$\mathbb{E}[\mathbf{X}_2 \mid \mathbf{X}_0 = \mathbf{x}] \;=\; \mathbf{x} + \tau \sum_{k=1}^{d} \boldsymbol{\zeta}_k \left[a_k(\mathbf{x}) + \mathbb{E}[a_k(\mathbf{X}_1 \mid \mathbf{X}_0 = \mathbf{x})]\right],$$

but when a_k is nonlinear, the quantity in the last expectation is a nonlinear function of a random vector. Extending to more steps leads to even more complicated embedded conditional expectations. This motivated us to try just the OSLAIF rule as a heuristic, and we got some good results with that. Specific illustrations of this are given in Section 6.4.

Let $\tilde{h}_j$ denote the functions h_j estimated from data as just described. These $\tilde{h}_j$ are noisy estimates, and since they are estimated separately across values of j, we can observe some random variation when looking at their sequence as a function of j. To smooth out this variation, we tried fitting a (least-squares) smoothing spline (de Boor, 2001; Pollock, 1993) to this sequence of functions $\tilde{h}_j$ to obtain a sequence of functions h_j, $j = 1,...s$, that varies more smoothly across the step number j. This yields a *smoothed SDIF*. In our experiments, we never observed a large improvement by doing this, because with $n = 2^{19}$ pilot simulations, the $\tilde{h}_j$ did not vary much already as a function of j. But the smoothing might be worthwhile when the number n of pilot simulations is smaller.

6.3.3. RQMC Point Sets

The RQMC point sets considered in this paper are the following (the short names in parentheses are used to identify them in the next section): (1) a randomly-shifted rank-1 lattice rule (Lat+s); (2) a Lat+s with the baker's transformation applied to the points after the shift (Lat+s+b); (3) a Sobol' net with a left random matrix scramble followed by a random digital shift (Sob+LMS); (4) a Sobol' net with the nested uniform scramble of Owen (1997) (Sob+NUS). These point sets and randomizations are defined and explained in L'Ecuyer (2009, 2018); L'Ecuyer and Lemieux (2000); Owen (2003). They are implemented

in SSJ (L'Ecuyer, 2016; L'Ecuyer and Buist, 2005), which we used for all our experiments. For the lattice rules, the parameters were found with the Lattice Builder tool (L'Ecuyer and Munger, 2016), using the weighted $\mathcal{P}_2$ criterion with order dependent weights of the form ρ^k, $\rho > 0$ for each projection of order k, for each k, with $\rho = 0.6$ for Example 6.4.1 and for the PKAr case in Example 6.4.3 (the small-dimensional cases), and $\rho = 0.05$ in all the other cases. The baker's transformation stretches each coordinate of each point by a factor or 2, then folds back the values by replacing u with $2 - u$ when $u > 1$. This is equivalent to transforming the integrand to make it periodic, which may improve the convergence rate (Hickernell, 2002) and may provide huge improvements in some cases, but it also increases the variation of the integrand, so it may also increase the variance (moderately) in other cases. For the Sobol' points, we used the parameters (direction numbers) from Joe and Kuo (2008), except for Example 6.4.1 and the PKAr case in Example 6.4.3, for which we used the parameters from Lemieux et al. (2004).

6.4. Numerical illustrations

We tested the performance of array-RQMC in combination with τ-leaping, first on some examples from Beentjes and Baker (2019), then on a higher-dimensional example from Padgett and Ilie (2016). All experiments were run using SSJ (L'Ecuyer, 2016; L'Ecuyer and Buist, 2005), which provides the required RQMC tools and also implements the sorting methods discussed in Section 6.3.2. The MRG32k3a random number generator was used for MC and all the randomizations.

We repeated each Array-RQMC procedure $m = 100$ times independently to estimate the RQMC variance $\mathrm{Var}[\hat{\mu}_n]$ for $n = 2^{13}, \ldots, 2^{19}$. We then fitted a model of the form $\mathrm{Var}[\hat{\mu}_n] \approx \kappa n^{-\beta}$ to these observations by least-squares linear regression in log-log scale. This gave an estimated convergence rate of $\mathcal{O}(n^{-\hat{\beta}})$ for the variance, where $\hat{\beta}$ is the least-square estimate of β. We report this $\hat{\beta}$ in our results. Ordinary MC gives $\beta = 1$, so we can compare. We also provide a few plots of $\mathrm{Var}[\hat{\mu}n]$ as a function of n, in log-log scale, to illustrate the typical behavior. Our logs are always in base 2, because we always use powers of 2 for n.

We computed the estimated *variance reduction factor* (VRF) of Array-RQMC compared with MC, which is defined as $\mathrm{Var}[g(\mathbf{X}_s)]/(n\mathrm{Var}[\hat{\mu}_n])$ where $\mathrm{Var}[g(\mathbf{X}_s)]$ is the MC variance for a single run, which was estimated separately by making $n = 2^{19}$ independent runs. This

is the variance per run for MC divided by the variance per run for Array-RQMC. We call VRF19 this value for $n = 2^{19}$ and we report it in our results. We also computed an efficiency ratio which measures the change in the work-normalized variance (the product of the estimator's variance by its computing cost). It is the VRF multiplied by the CPU time required to compute n realizations with MC and divided by the CPU time to compute the RQMC or Array-RQMC estimator with the samen. We call EIF19 its value for $n = 2^{19}$ and we report it as well. This measure takes into account both the gain in variance and the extra cost in CPU time which is required to sort the chains at each step of the Array-RQMC algorithm. Note that using RQMC only is generally not slower than MC, but usually a bit faster.

6.4.1. Reversible isomerization system

We start with the same simple model of a *reversible isomerization system*, taken from Beentjes and Baker (2019). There are two species, S_1 and S_2, and $d = 2$ reaction channels with reaction rates $c_1 = 1$ and $c_2 = 10^{-4}$:

$$S_1 \underset{c_2}{\overset{c_1}{\rightleftarrows}} S_2.$$

We start with $X_1(0) = 10^2$ molecules of type S_1 and $X_2(0) = 10^6$ molecules of type S_2. Since the total number of molecules is constant over time, it suffices to know the number of molecules of the first type, $X_1(t)$, at any time t, so we can define the state of the CTMC as $X(t) = X_1(t)$ only. This gives us $\ell = 1$. Then, we only need a one-dimensional sort for Array-RQMC. We also take $g(X(t)) = X_1(t)$. Note that with our choice of initial state, $\mathbb{E}[X_1(t)] = 10^2$ for all $t > 0$, so we already know the answer for this simple example. There are two possible reactions, so $d = 2$, and we therefore need RQMC points in $2s$ dimensions with classical RQMC and in $\ell + d = 3$ dimensions with Array-RQMC.

Table 6.1 summarizes our experimental results. Seven cases are reported in the table. The first case (in the upper left) has the same parameters as Beentjes and Baker (2019): $T = 1.6$, and $s = 8$, so $\tau = T/s = 0.2$. Figure 6.1 displays how the variance decreases as a function of n for this case. Notice the steeper slope for the four Array-RQMC variants. Array-RQMC clearly outperforms both MC and classical RQMC in this example.

We also observe with these three cases that when we increase s with T fixed, the factors VRF19 and EIF19 diminish, and the diminution is much more important with RQMC. The

latter might be no surprise, because increasing s increases the dimension of the RQMC points. But it was unclear a priori if it would also occur with Array-RQMC, and how much. However, by doing further experimentation, we found that the decrease of VRF19 is not due really to the increase in the number of steps, but rather to the decrease in τ. To see that, look at the fourth case, with $(T,s,\tau) = (25.6,128,0.2)$. Here we have the same τ as in the first case, but s is multiplied by 16. For the Array-RQMC methods, the variance reductions and convergence rates are similar to the first case. For RQMC, they are a bit lower, which is not surprising because the dimension has increased. For cases five and six, we have increased τ to 0.8 and we compare two large values of s. The VRF19's are roughly comparable, which means that they really depend on τ and not much ons. Why is that?

Recall that in this example, at each step we generate a pair of Poisson random variables, which are discrete and therefore discontinuous with respect to the underlying uniforms. The mean of each Poisson random variable is proportional to τ, and the larger the mean, the closer it is to a continuous distribution. In fact, as τ increases, the Poisson converges to anormal distribution, whose inverse cdf is smooth, so the generated values are smooth functions of the underlying uniforms in the limit. That is, we obtain a better VRF19 when τ is larger because the integrand is closer to a continuous (and smooth) function. When the Poisson distributions have small means, in contrast, the response has larger discontinuities. And it is well known that RQMC is much more effective for smooth functions that discontinuous functions. This kind of behavior was already pointed out for RQMC in Section 5.2 of Beentjes and Baker (2019). Interestingly, we see that the same effect applies to Array-RQMC as well. To illustrate this effect "in the limit", we made an experiment in which all the Poisson random variables at each step are replaced by normals with the same mean and variance, and the state vector has real-valued components rather than integer components, using the same parameters as in the first case in the table. The results are in the last (bottom) entry of the table and they are stunning. Firstly, for RQMC and all Array-RQMC methods, the rate $\hat{\beta}$ is close to 2, which does not occur for the other cases. Secondly, the VRF19 factor is also very large for RQMC and is huge in particular for Array-RQMC with Lat+s+b. This surprising result for RQMC can be explained as follows. Here the integrand has 16 dimensions, but on a closer look one can see that it is a sum of 16 normal random variables that are almost independent; i.e., almost a sum of one-dimensional functions. This

means that the effective dimension is close to 1, and this explains the success of RQMC. Regarding the huge gain with Lat+s+b, it can be explained by the fact that for a smooth one-dimensional function, RQMC with Lat+s+b can provide an $\mathcal{O}(n^{-4})$ convergence rate for the variance (Hickernell, 2002; L'Ecuyer, 2009). Essentially, for one-dimensional smooth functions, the baker's transformation produces a locally antithetic effect which integrates exactly the piecewise linear approximation, and only higher-order error terms remain. The huge VRF19 indicates that part of this effect carries over to Array-RQMC.

We just saw that as a rough rule of thumb, the RQMC methods bring more gain when the Poisson random variables have larger means. We know (from Section 6.2) that the mean of the Poisson random variable $D_{j,k}$ is $a_k(\mathbf{X}_{j-1})\tau$. This mean can be increased by increasing either τ or the components of the state vector. For the present example, if we denote $\mathbf{X}_{j-1} = (X_{j-1}^{(1)}, X_{j-1}^{(2)})^t$, the number of molecules of each of the two types at step $j-1$, we have $a_k(\mathbf{X}_{j-1}) = c_k X_{j-1}^{(k)}$ for $k = 1,2$, so the Poisson means are increased by a factor $\gamma > 1$ by either multiplying τ by γ or multiplying the vector $\mathbf{X}_{j-1}$ by γ. We made experiments whose results agreed with that when all the components of the state were large enough. But if one component of $\mathbf{X}_{j-1}$ is small, and we increase τ and simulate the system over a few steps, this component has a good chance of getting close to zero at some step, and this increases the discontinuity. In that situation, a larger τ can worsen the VRF. To further test the above reasoning, we made another set of experiments in which the initial state $\mathbf{X}_0$ had two equal components, exactly $X_0^{(1)} = X_0^{(2)} = (10^2 + 10^6)/2$ molecules of each type, and we adapted the reaction rates to $c_1 = c_2 = 100/X_0^{(1)}$, to keep $\mathbb{E}[X_1(t)] = X_0^{(1)}$ for all t. In this case, the problem of one component getting close to 0 does not occur so things remain smoother. We found that the VRFs were larger than in Table 6.1 for both RQMC and Array-RQMC (we exclude the normal distribution). The VRF for RQMC was also smaller when both T and s were large, but not when s was increased and T remained small. One possible explanation for this is that when T and s are large, the overall change in the state can be large, and then the set of successive changes in the state are less independent, which increases the effective dimension.

(T,s,τ) $\longrightarrow$	(1.6,8,0.2)			(1.6,1024,0.2/128)			(1.6,1024,0.2/128)		
MC Var	107.8			96.6			96.0		
Sample	$\hat{\beta}$	VRF19	EIF19	$\hat{\beta}$	VRF19	EIF19	$\hat{\beta}$	VRF19	EIF19
MC	1.00	1	1	1.00	1	1	1.00	1	1
RQMC	1.03	629	1,493	1.08	79	83	1.01	46	68
Lat+s	1.81	27,864	14,835	1.82	17,163	5,315	1.64	6,918	2,098
Lat+s+b	1.60	14,420	7,636	1.7	46,133	1,838	1.53	1,997	553
Sob+LMS	1.64	18,892	9,819	1.62	8,370	2,785	1.57	3,949	1,386
Sob+NUS	1.70	17,518	6,440	1.60	5,773	1,095	1.65	3,182	667
(T,s,τ) $\longrightarrow$	(25.6,128,0.2)			(102.4,128,0.8)			(819.2,1024,0.8)		
MC Var	111.0			166.7			166.6		
Sample	$\hat{\beta}$	VRF19	EIF19	$\hat{\beta}$	VRF19	EIF19	$\hat{\beta}$	VRF19	EIF19
MC	1.00	1	1	1.00	1	1	1.00	1	1
RQMC	1.06	519	625	1.10	2,294	2,381	1.12	2,887	3,018
Lat+s	1.78	21,084	11,030	1.83	32,538	23,379	1.81	31,909	23,372
Lat+s+b	1.80	26,566	13,666	1.72	45,841	33,229	1.66	47,264	35,627
Sob+LMS	1.62	17,358	9,290	1.66	46,883	32,813	1.49	32,220	23,094
Sob+NUS	1.61	15,624	6,044	1.57	40,161	23,552	1.53	30,970	17,123

(T,s,τ) $\longrightarrow$	(1.6,8,0.2), normal		
MC Var	107,8		
Sample	$\hat{\beta}$	VRF19	EIF19
MC	1.00	1	1
RQMC	1.14	11	12
Lat+s	1.55	1814	1051
Lat+s+b	1.26	4375	2567
Sob+LMS	1.37	6133	3775
Sob+NUS	1.33	5254	2813

Table 6.1. Estimated rates $\hat{\beta}$, VRF19, and EIF19, for the reversible isomerization example, for various choices of (T,s,τ). MC refers to ordinary MC, RQMC is classical RQMC with Sobol' points and LMS randomization, and the other four rows are for Array-RQMC with different RQMC point sets. "MC Var" is $\mathrm{Var}[g(\mathbf{X}_s)]$, the variance per run with MC.

6.4.2. Schlögl system

In this second example, also taken from Beentjes and Baker (2019), we have the three species S_1, S_2 and S_3, and four reaction channels with reaction rates $c_1 = 3\cdot 10^{-7}$, $c_2 = 10^{-4}$, $c_3 = 10^{-3}$ and $c_4 = 3.5$, respectively. The model can be depicted as:

$$2S_1 + S_2 \underset{c_2}{\overset{c_1}{\rightleftarrows}} 3S_1,$$

$$S_3 \underset{c_4}{\overset{c_3}{\rightleftarrows}} S_1.$$

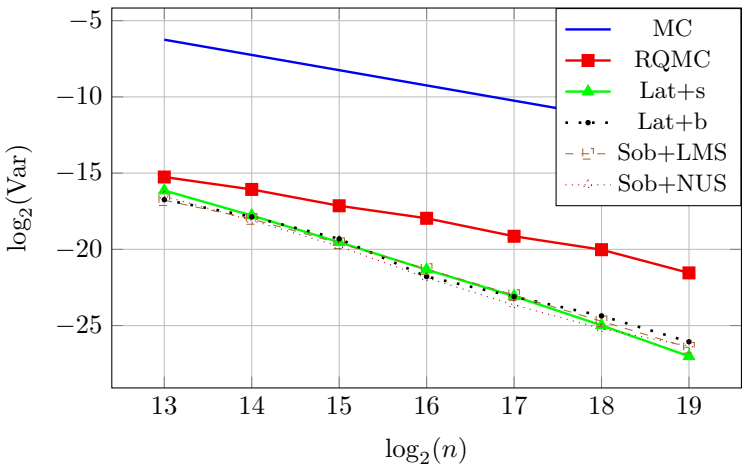

Figure 6.1. Estimated $\mathrm{Var}[\hat{\mu}_n]$ as a function of n, in log-log scale, for the reversible isomerization system, with $T = 1.6$ and $s = 8$.

The propensity functions a_k are given by

$$a_1(\mathbf{x}) = c_1 x_1(x_1 - 1)x_2/2, \qquad a_2(\mathbf{x}) = c_2 x_1(x_1 - 1)(x_1 - 2)/6,$$
$$a_3(\mathbf{x}) = c_3 x_3, \qquad a_4(\mathbf{x}) = c_4 x_1.$$

We also take $\mathbf{x}_0 = \{250, 10^5, 2 \cdot 10^5\}$, $T = 4$, and $\tau = T/16$, so $s = 16$ steps. This is the same model as in Beentjes and Baker (2019), with the same parameters, except that we took a smaller τ to avoid negative copy numbers. We want to estimate $\mathbb{E}[X_1(T)]$, the expected number of molecules of S_1 at time T. Here, this expectation does depend on T, and we will see that $\mathrm{Var}[X_1(T)]$ also depends very much on T.

Since the total number of molecules remains constant over time, the dimension of the state here can be taken as $\ell = 2$. We take the state as $\mathbf{X} = (X^{(1)}, X^{(2)})^t$, and $X^{(3)}$ can be deduced by $X^{(3)} = N_0 - X^{(1)} - X^{(2)}$ where N_0 is the total number of molecules. With $d = 4$ possible reactions, the RQMC points must be five-dimensional if we construct an importance function h that maps the state to one dimension, and must be six-dimensional otherwise. For comparison, with classical RQMC, the dimension of the RQMC points is $d\lceil T/\tau \rceil = 64$.

In the previous example, the state was one-dimensional, so there was no need to define an importance function for Array-RQMC, but here we have a two-dimensional state. We now examine how to construct an importance function $h_j : \mathbb{N}_0^2 \to \mathbb{R}$ as discussed in Section 6.3.2. To construct an importance function using the OSLAIF, when $g(\mathbf{x})$ is a linear function

of $\mathbf{x}$, one can compute the conditional expectation *exactly* by using (6.3.2). This gives a polynomial of the form:

$$h_j(x_1,x_2) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_1 x_2 + \theta_5 x_1^3 + \theta_6 x_1^2 x_2. \tag{6.4.1}$$

With $g(\mathbf{x}) = x_1$ (our case), the coefficients are the following: $(\theta_0, \theta_1, \ldots, \theta_6) = (300.25\tau, 1 - 3.5\tau, -10^{-3}\tau, 5 \times 10^{-5}\tau, -1.5 \times 10^{-7}\tau, -1.67 \times 10^{-5}\tau, 1.5 \times 10^{-7}\tau)$. When x_1 is very large, we can approximate $a_1(\mathbf{x}) \approx c_1 x_1^2 x_2/2$ and $a_2(\mathbf{x}) = c_2 x_1^3/6$, and then remove the two terms $\theta_3 x_1^2 + \theta_4 x_1 x_2$ from (6.4.1), but in our example, x_1 is not very large.

To obtain a SDIF for a more general j, one possible heuristic is to assume the same form of polynomial (even if this is not exact) and select the coefficients θ_i by least-squares fitting to data obtained $n = 2^{19}$ pilot runs as explained in Section 6.3.2. We did this and we also tried fitting a more general bivariate polynomial that contains all possible monomials $x_1^{\varepsilon_1} x_2^{\varepsilon_2}$ with $0 \le \varepsilon_1, \varepsilon_2 \le 3$, but this gave us no improvement over OSLAIF. The other SDIF approches that we tried also did no better than OSLAIF. A plausible explanation is that the functions h_j in this case are based on data obtained from noisy simulations (large variance and dependence on j). A possible alternative could be to use automatic learning with a deep neural network to learn a good h. But this is beyond our scope.

		$T = 4$, $s = 16$			$T = 4$, $s = 128$			$T = 32$, $s = 128$		
MC Var		27,409			27,471			270		
Sort	Sample	$\hat{\beta}$	VRF19	EIF19	$\hat{\beta}$	VRF19	EIF19	$\hat{\beta}$	VRF19	EIF19
	MC	1.00	1	1	1.00	1	1	1.00	1	1
	RQMC	1.14	11	12	1.04	7	8	1.29	211	203
OSLAIF	Lat+s	1.55	1814	1051	1.49	2072	1403	1.23	508	541
	Lat+s+b	1.26	4375	2567	1.38	1230	861	1.08	471	506
	Sob+LMS	1.37	6133	3775	1.46	3112	2285	1.11	556	629
	Sob+NUS	1.33	5254	2813	1.49	3258	1556	1.13	461	483
Batch	Lat+s	1.62	2979	2657	1.58	2150	830	1.47	2136	2259
	Lat+s+b	1.39	4831	2650	1.39	1123	415	1.34	1682	1791
	Sob+LMS	1.54	8147	4880	1.46	2202	1503	1.26	1614	1274
	Sob+NUS	1.44	5761	3007	1.42	2024	1102	1.26	1483	989
Hilbert	Lat+s	1.46	509	419	1.41	652	350	1.33	837	614
	Lat+s+b	1.49	1468	1213	1.18	375	216	1.28	1159	848
	Sob+LMS	1.58	3604	1973	1.29	575	313	1.25	1642	1231
	Sob+NUS	1.58	3183	1657	1.28	617	255	1.28	1358	881

Table 6.2. Estimated variance rates $\hat{\beta}$, EIF19 and VRF19 for the Schlögl system, with various types of sorts for Array-RQMC.

For the batch sort, we kept the three coordinates in their natural ordering and we used . $n_1 = \lceil n^{1/2} \rceil$ and $n_2 = \lceil (n/n1)^{3/8} \rceil$. For $n = 2^{19}$, this gives $n_1 = 725$, $n_2 = 27$, and $n_3 = 2$.

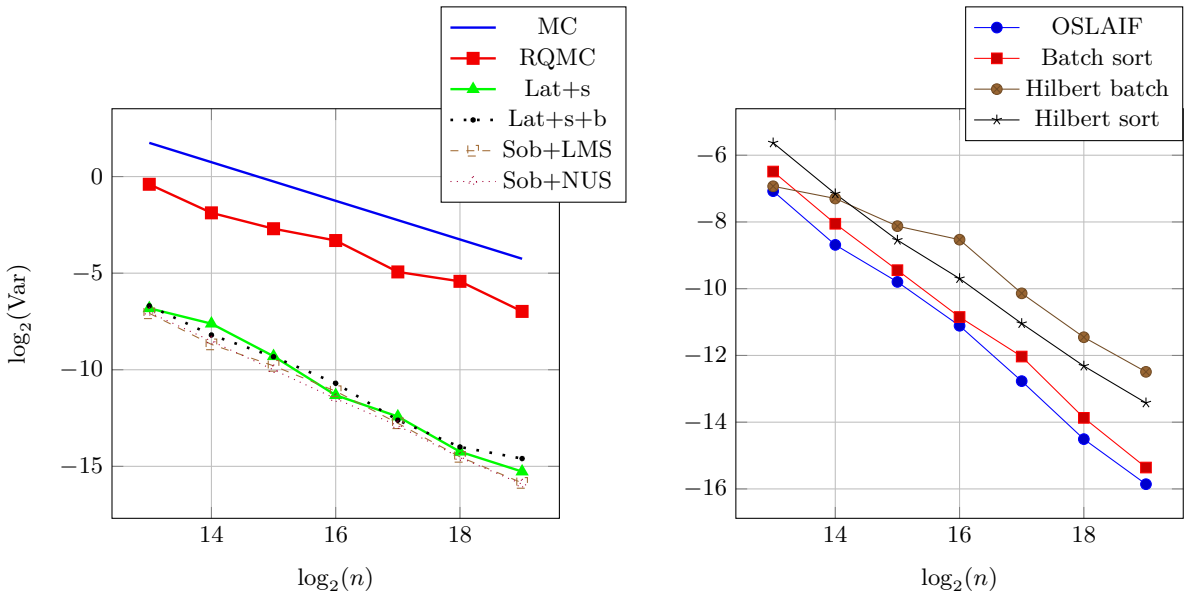

Figure 6.2. Empirical variance of the sorting methods vs n in a log-log scale, T=4, s=128, for the OSLAIF sort and various point sets (left) and for various sorts with Sobol+LMS (right).

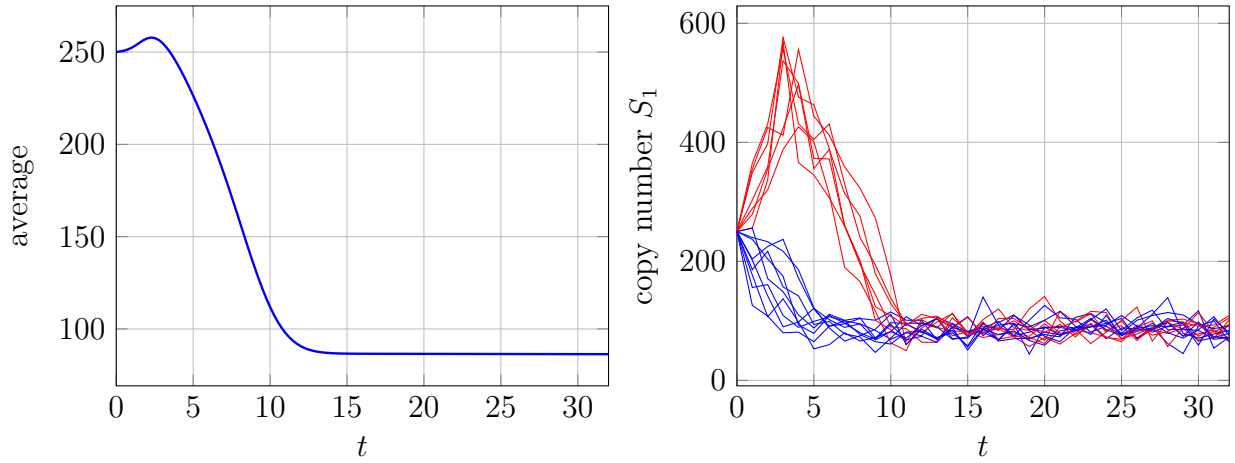

Figure 6.3. The mean with $n = 2^{19}$ (left) and the trajectories of $X_1(t)$ for $n = 16$ chains for $t \leq 32$ (right).

Table 6.2 summarizes our experimental results with this example. Array-RQMC performs much better than RQMC. It reduces the variance by factors in the thousands, and over 8,000 in one case with $n = 2^{19}$. The OSLAIF, Hilbert curve sort, and batch sort all perform reasonably well, which is not very surprising, because the state space is only two-dimensional. The OSLAIF is very effective for $T = 4$, but somewhat less effective for $T = 32$. Globally, the batch sort is the best performer; its VRF19 and EIF19 values are both consistently among the largest ones. The Sobol' points are generally the best performers for each type of sort.

The left panel of Figure 6.2 shows $\mathrm{Var}[\hat{\mu}_n]$ under Sob+LMS as a function of n in a log-log-scale. The right panel shows $\mathrm{Var}[\hat{\mu}_n]$ versus n in log-log scale for the OSLAIF sort, for various point sets. The estimated convergence rates $\hat{\beta}$ are mostly between 1.3 and 1.6, which clearly beats the usual MC rate of 1 and the classical RQMC rate of 1.15. The right panel shows $\mathrm{Var}[\hat{\mu}_n]$ as a function of n under Sob+LMS, in a log-log-scale.

One important observation is the large difference in MC variance between $T = 4$ and $T = 32$; it is larger at $T = 4$ by a factor of about 100. The mean $\mathbb{E}[\hat{\mu}_n]$ also depends on T: it is about 240 at $T = 4$ and about 86 at $T = 32$. It is plotted as a function of T in the left panel of Figure 6.2. What happens is that the trajectories have roughly two very different kinds of transient regimes between $t = 0$ and about $t = 10$. For some trajectories, $X_1(t)$ goes up to somewhere between 400 and 600 at around $t = 4$, then goes down to around the long-term mean, say between 70 and 100. For other trajectories, $X_1(t)$ decreases right away to between 70 and 100 at around $t = 5$. Figure 6.3 illustrates this behavior, with 16 sample paths. This was already mentioned in Beentjes and Baker (2019), although they say the system is bistable, whereas what we observe is rather two types of transient paths. This behavior explains the much larger variance at $T = 4$ than at $T = 32$. It also shows why it is very hard to predictthe state at some larger T from the state at $t = 1/4$, say, hence the difficulty to estimate an "optimal" importance function. Despite all of this, Array-RQMC performs quite well with simple sorts and brings large efficiency improvements compared with MC and RQMC.

6.4.3. The cyclic adenosine monophosphate activation of protein kinase A model

This example is a model for the cyclic adenosine monophosphate (cAMP) activation of protein kinase A (PKA), taken from Koh and Blackwell (2012) and Strehl and Ilie (2015). This model is interesting because it has $\ell = 6$ and $d = 6$, which are both larger than in the previous examples. The six molecular species S_1 to S_6 are (in this order) PKA, cAMP, the partially saturated PKA-cAMP$_2$, the saturated PKA-cAMP$_4$, the regulatory subunit PKAr,

and the catalytic subunit PKAc. The $d = 6$ possible reactions are depicted here:

$$\text{PKA} + 2\text{cAMP} \underset{c_2}{\overset{c_1}{\rightleftarrows}} \text{PKA-cAMP}_2,$$

$$\text{PKA-cAMP}_2 + 2\text{cAMP} \underset{c_4}{\overset{c_3}{\rightleftarrows}} \text{PKA-cAMP}_4,$$

$$\text{PKA-cAMP}_4 \underset{c_6}{\overset{c_5}{\rightleftarrows}} \text{PKAr} + 2\text{PKAc}.$$

The propensity functions a_k are given by

$$a_1(\mathbf{x}) = c_1 x_2(x_2 - 1)x_1/2, \qquad a_2(\mathbf{x}) = c_2 x_3,$$

$$a_3(\mathbf{x}) = c_3 x_2(x_2 - 1)x_3/2, \qquad a_4(\mathbf{x}) = c_4 x_4,$$

$$a_5(\mathbf{x}) = c_5 x_4, \qquad a_6(\mathbf{x}) = c_6 x_6(x_6 - 1)x_5/2.$$

The reaction rates are $c_1 = 2.6255 \times 10^{-6}$, $c_2 = 0.02$, $c_3 = 3.8481 \times 10^{-6}$, $c_4 = 0.02$, $c_5 = 0.016$ and $c_6 = 5.1325 \times 10^{-5}$. We simulate this system with the same parameters as Padgett and Ilie (2016), except that we assume that the molecules are homogeneously distributed in the volume and we choose a fixed τ as opposed to selecting it adaptively after each step. At time zero there are 33,030 molecules of cAMP, 33,000 molecules of PKA, and 1,100 molecules of each other species. We take $T = 0.05$ and $\tau = T/256$, so $s = 256$ steps. This problem requires RQMC points in 7 or 12 dimensions with array-RQMC, compared with 1536 dimensions with classical RQMC.

We are reporting experiments with two different objective functions here. The first one is $\mathbb{E}[X_1(T)]$, the expected number of molecules of PKA at time T, and the second one is $\mathbb{E}[X_5(T)]$, the expected number of molecules of PKAr at time T. In each case, we implemented and tested the OSLAIF and SDIF methods to select a mapping h to one dimension. We also tried the multivariate batch and split sorts, the Hilbert curve sort, and the Hilbert batch sort, from Section 6.3.2. The best performers were the OSLAIF map, the batch sort, and the Hilbert sort.

We first observe PKA, for which $g(\mathbf{x}) = x_1$. The OSLAIF in this case is given by the polynomial $h_j(\mathbf{x}) = x_1 + \tau(-c_1 x_1 x_2(x_2 - 1)/2 + c_2 x_3)$. In this function, the magnitude x_1 outweighs that of the $-\tau c_1 x_1 x_2(x_2 - 1)/2$ term on average, followed by $\tau c_2 x_3$. his suggests taking x_1 as the most important coordinate for the sort, followed by x_2 and x_3. So for the

batch sort, we used the three coordinates x_1, x_2, x_3 in this order. We tried a few settings for the batch sizes and ended up with $n_1 = \lceil n^{1/2} \rceil$, $n_2 = \lceil (n/n_1)^{3/8} \rceil$, and $n_3 = \lceil (n/n_1/n_2)^{1/8} \rceil$. For $n = 2^{19}$, this gives $n_1 = 725$, $n_2 = 12$, and $n_3 = 2$.

Sort	Sample	$\hat{\beta}$	VRF19	EIF19
	MC	1.00	1	1
	RQMC	1.08	464	603
OSLAIF	Lat+s	1.50	1420	806
	Lat+s+b	1.26	745	423
	Sob+LMS	1.29	1295	937
	Sob+NUS	1.29	1174	480
Batch	Lat+s	1.43	116	356
	Lat+s+b	1.25	1264	604
	Sob+LMS	1.15	1699	981
	Sob+NUS	1.22	1633	353
Hilbert	Lat+s	1.27	1181	452
	Lat+s+b	1.06	821	280
	Sob+LMS	1.15	850	327
	Sob+NUS	1.24	1217	266

Table 6.3. Estimated rates $\hat{\beta}$, VRF19, and EIF19 for PKA with $T = 0.05$, $s = 256$.

Sort	Sample	$\hat{\beta}$	VRF19	EIF19
	MC	1.03	1	1
	RQMC	1.17	39	45
By PKAc	Lat+s	1.33	2470	1585
	Lat+s+b	1.36	1364	1248
	Sob+LMS	1.45	1856	1580
	Sob+NUS	1.50	2053	668
OSLAIF	Lat+s	1.42	3634	1550
	Lat+s+b	1.38	1491	627
	Sob+LMS	1.47	2062	1059
	Sob+NUS	1.51	2184	712
Batch-5	Lat+s	1.48	225	116
	Lat+s+b	1.49	406	203
	Sob+LMS	1.43	576	366
	Sob+NUS	1.37	668	180
Batch-6	Lat+s	1.62	3026	1560
	Lat+s+b	1.43	1592	796
	Sob+LMS	1.46	2753	1749
	Sob+NUS	1.47	2486	670
Hilbert	Lat+s	1.17	135	54
	Lat+s+b	1.12	88	27
	Sob+LMS	1.24	126	60
	Sob+NUS	1.22	161	50

Table 6.4. Estimated rates $\hat{\beta}$, VRF19, and EIF19 for PKAr with $T = 0.05$, $s = 256$.

Table 6.3 summarizes our results. The estimated mean and variance per run are also 19663 and 1775, respectively. We find that the three sorting methods reported in the table

offer comparable performance in terms of VRF19, although OSLAIF and the batch sort dominate when we look at the EIF19. This is because sorting on a single value or a restricted set of coordinates, as we do for the batch sort, is faster than a full multivariate sort. Classical RQMC also performs surprisingly well despite the large number of dimensions, but not as well as Array-RQMC with the best sorts. With Array-RQMC, we also observe empirical convergence rates $\hat{\beta}$ consistently better than the MC rate of 1.0. This indicates that the VRF should increase further with n.

Table 6.4 gives the results for the PKAr case, for which $g(\mathbf{x}) = x_5$. The estimated mean and variance per run are about 716 and 47, respectively. For this case, the OSLAIF is given by $h(\mathbf{x}) = x_5 + \tau(c_5 x_4 - 0.5 c_6 x_5 x_6 (x_6 - 1))$. Given that x_4, x_5, and x_6 remain roughly between 500 and 1000 in this model, and that $\tau = 1/5120$, the dominating term in this function is (by far) x_5, followed by $-\tau c_6 x_5 x_6^2 \approx -2.5 \times 10^{-3} x_5$. Based on this, for the batch sort, we initially used the coordinates x_5, x_6, x_4 in this order, and took $n_1 = \lceil n^{1/2} \rceil$, $n_2 = \lceil (n/n_1)^{3/8} \rceil$, and $n_3 = \lceil (n/n_1/n_2)^{1/8} \rceil$ for the batch sizes, as in the previous case. This is denoted by "Batch-5" in the table.

We also tried SDIF with various types of functions, but it did not really perform better. While doing that, we applied the random forest permutation-based statistical procedure of Breiman (2001) to detect the most important variables in a noisy function. This procedure told us that x_6 was by far the most important variable for the sort, at all steps. Therefore, we also tried sorting the states by the number of PKAc molecules only. We call this sorting method "By PKAc" in Table 6.4.

The OSLAIF, Batch-6, and "By PKAc" sorts perform similarly. They provide large improvement factors for both the variance and the efficiency, and empirical convergence rates $\hat{\beta}$ that are significantly larger than 1. Their performance is orders of magnitude better than RQMC. The Batch-5 and Hilbert sorts are not competitive with the other ones in this case, but they nevertheless reduce the variance by significant factors.

This example illustrates two facts. First, the dimension of the state is not the ultimate criterion for Array-RQMC to perform well. Secondly, customizing sorting algorithms based on information on the underlying model can improve results significantly.

6.5. Conclusion

We have studied the combination of the fixed-step τ-leap algorithm with array-RQMC for well-mixed chemical reaction networks and found that in this way, we can reduce the variance in comparison to MC significantly. In contrast to the simulation with traditional RQMC, this approach could often also improve the convergence rate of the variance. Array-RQMC requires to sort the chains by their states at each step of the chain with a multivariate sort, which can become costly when the state space has large dimensionality. We also showed that one can construct sorts by mapping the states into the real numbers by an uncomplicated importance function, where sorting is trivial. A simple variant named OSLAIF performs comparably as well or better than several standard sorting algorithms, while being naturally easier and less costly to apply. We have also shown that obtaining additional knowledge of the model, such as identifying important variable projections, and adapting a sort to this information can boost the convergence of the variance tremendously, while the standard multivariate sorts might not capture this information well at all.

ACKNOWLEDGMENTS

This work has been supported by a Canada Research Chair, an IVADO Research Grant, and an NSERC Discovery Grant number RGPIN-110050 to P. L'Ecuyer. F. Puchhammer was also supported by Spanish and Basque governments fundings through BCAM (ERDF, ESF, SEV-2017-0718, PID2019-108111RB-I00, PID2019-104927GB-C22, BERC 2018e2021, EXP. 2019/00432, ELKARTEK KK-2020/00049), and the computing infrastructure of i2BASQUE academic network and IZO-SGI SGIker (UPV).

Chapter 7

Conclusion and Future Research Perspectives

This thesis is based on a collection of four articles. The first one was accepted for publication in a journal, the second one, is currently under revision in an international journal, the third one was published in international conference and the last one has been submitted for possible publication. In this chapter we summarize the main results and directions for future research.

7.1. Conclusion

In this thesis, we first investigated the combination of RQMC and stratification with two common density estimators, namely histograms and kernel density estimators (KDE) to estimate a density by simulation. We showed that using RQMC or stratification, the IV and the MISE can be reduced especially when the dimension is small. The improvement is generally more limited when the dimension is large. We also found that if the bandwidth $h \to 0$, the IV enhancement rapidly degrades as a function of h. In our empirical experiments, using RQMC or stratification, the IV for both density estimators is never considerably larger when using MC, and is often much smaller.

Secondly, We examined a novel strategy to estimate by conditioning the density of a random variable generated by simulation from a stochastic model. The resulting conditional density estimator (CDE) is unbiased and consequently its MISE converges at a faster rate than for other common density estimators like the KDE. By combining the CDE with RQMC sampling, we have also shown how to further decrease the IV and even enhance its convergence rate. Our numerical examples demonstrate that this combination can be very efficient and it can sometimes reduces the MISE by factors over a million in some examples.

Thirdly, we showed how array-RQMC can be applied for pricing options under stochastic volatility models, and provided thorough examples with the Variance Gamma, Heston, and Ornstein-Uhlenbeck models. With these models, the method requires higher-dimensional RQMC points than with the simpler Geometric Brownian Motion model previously studied, and when the time has to be discretized to apply Euler's method, the number of Markov chain steps is much larger. Our empirical results show that, compared to the crude Monte Carlo, the variance can be significantly reduced.

Finally, we studied the combination of the fixed-step τ-leap algorithm with array-RQMC for well-mixed chemical reaction networks and found that in this way, the variance can be significantly reduced in comparison to MC. This strategy could also often enhance the convergence rate of the variance as opposed to the simulation with traditional RQMC. At each step of the chain, array-RQMC requires a multivariate sorting of the chains by their states, which can become costly when the state space has a large dimension. We also showed that one can construct sorts by mapping the states into the real numbers by an uncomplicated importance function and sort the states according to this importance function. These customized sorts performed comparably well or even better than several standard sorting algorithms but they are naturally easier to apply and therfore less costly. We have also shown that acquiring extra model understanding, such as identifying significant variable projections, and adjusting a sort to this information can extremely enhance the convergence of the variance, while the standard multivariate sorts may not capture this information well at all.

7.2. Future Research

In the following we present future work, starting with four ongoing research that have not been presented in the thesis. The first ongoing research is related to the combination of RQMC for density estimation with Conditional Monte Carlo. The second ongoing work concerns the sensitivity estimator which is derived from a simple application of Likelihood Ratio method, that is typically used for derivative estimation of performance measures in Discrete Event Systems. The last two future research concern the variance estimation and density estimation by array-RQMC method.

Suggested future work includes experimenting the combination of RQMC with the conditional density estimator on larger and more complicated stochastic models, designing and exploring different types of conditioning, and perhaps adapting the Monte Carlo sampling strategies (e.g., changing the way X is defined in terms of the basic input random variates) to make the method more effective. In addition, it will be interesting to investigate the combination of the stratified sampling with this estimator and give proofs for the convergence rate of the IV and compare it with the ones obtained by the combination with the RQMC poinst set for each example.

A different (but related) approach called sensitivity estimator (SE), was proposed by Laub et al. (2019). It combines a clever change of variable with the likelihood ratio method for derivative estimation, to estimate the density of a sum of dependent random variables. An interesting direction for future work is to extend their approach to a far more general setting and applied it in applications where the conditional density estimator cannot be effective.

For variance estimation by array-RQMC, so far, L'Ecuyer et al. (2008) have provided some convergence proofs for special cases where the dimension of the state is $l = d = 1$ and under some conditions on φ. It will be interesting to extend these proofs for the case where $l > 1$ and $d > 1$, and apply this method to other examples and applications in a high dimensionnal space.

We also plan to extend the array-RQMC method to estimate density by using histograms or kernel density. Furthermore, we are interested in providing convergence proofs for the empirically-observed rates and investigating other strategies of sorting in order to deal with multidimensional states and in particular a potential solution would be to use automated learning strategies to find a good importance function that will be used to sort the states.

Bibliography

D. Anderson and D. Higham. Multilevel Monte Carlo for continuous-time Markov chains, with applications in biochemical kinetics. *Multiscale Modeling & Simulation*, 10(1):146–179, 2012. doi: 10.1137/110840546.

D. F. Anderson. Incorporating postleap checks in tau-leaping. *The Journal of Chemical Physics*, 128(5):054103, 2008. URL https://doi.org/10.1063/1.2819665.

D. F. Anderson and T. G. Kurtz. Continuous time Markov chain models for chemical reaction networks. In H. Koeppl, D. Densmore, G. Setti, and M. di Bernardo, editors, *Design and analysis of biomolecular circuits*, volume 117, pages 3–42. Springer, New York, 2011.

W. J. Anderson. *Continuous-Time Markov Chains: An Applications-Oriented Approach*. Springer-Verlag, New York, 1991.

S. Asmussen. Conditional Monte Carlo for sums, with applications to insurance and finance. *Annals of Actuarial Science*, 12(2):455–478, 2018.

S. Asmussen and P. W. Glynn. *Stochastic Simulation*. Springer-Verlag, New York, 2007.

A. Ault. An introduction to enzyme kinetics. *Journal of Chemical Education*, 51(6):381, 1974.

A. N. Avramidis and P. L'Ecuyer. Efficient Monte Carlo and quasi-Monte Carlo option pricing under the variance-gamma model. *Management Science*, 52(12):1930–1944, 2006.

A. N. Avramidis and J. R. Wilson. Integrated variance reduction strategies for simulation. *Operations Research*, 44:327–346, 1996.

A. N. Avramidis and J. R. Wilson. Correlation-induction techniques for estimating quantiles in simulation experiments. *Operations Research*, 46(4):574–591, 1998.

A. N. Avramidis, P. L'Ecuyer, and P.-A. Tremblay. Efficient simulation of gamma and variance-gamma processes. In *Proceedings of the 2003 Winter Simulation Conference*, pages 319–326, Piscataway, New Jersey, 2003. IEEE Press.

C. H. L. Beentjes and R. E. Baker. Quasi-Monte Carlo methods applied to tau-leaping in stochastic biological systems. *Bulletin of Mathematical Biology*, 81:2931–2959, 2019.

A. Ben Abdellah, P. L'Ecuyer, A. Owen, and F. Puchhammer. Density estimation by randomized quasi-Monte Carlo. manuscript, arXiv:1807.06133, 2019a.

A. Ben Abdellah, P. L'Ecuyer, and F. Puchhammer. Array-RQMC for option pricing under stochastic volatility models. In *Proceedings of the 2019 Winter Simulation Conference*, pages 440–451. IEEE Press, 2019b. URL https://www.informs-sim.org/wsc19papers/429.pdf.

A. Berlinet and L. Devroye. A comparison of kernel density estimates. *Publications de l'Institut de Statistique de l'Université de Paris*, 38(3):3–59, 1994.

D. Bingham. Virtual library of simulation experiments, 2017. URL https://www.sfu.ca/~ssurjano/canti.html.

Z. I. Botev, P. L'Ecuyer, G. Rubino, R. Simard, and B. Tuffin. Static network reliability estimation via generalized splitting. *INFORMS Journal on Computing*, 25(1):56–71, 2013.

Z. I. Botev, P. L'Ecuyer, and B. Tuffin. Static network reliability estimation under the Marshall-Olkin copula. *ACM Transactions on Modeling and Computer Simulation*, 26(2): Article 14, 28 pages, 2016.

P. Bratley, B. L. Fox, and L. E. Schrage. *A Guide to Simulation*. Springer-Verlag, New York, NY, second edition, 1987.

L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

R. E. Caflisch, W. Morokoff, and A. Owen. Valuation of mortgage-backed securities using Brownian bridges to reduce effective dimension. *J. of Computational Finance*, 1(1):27–46, 1997.

D. Choquet, P. L'Ecuyer, and C. Léger. Bootstrap confidence intervals for ratios of expectations. *ACM Transactions on Modeling and Computer Simulation*, 9(4):326–348, 1999.

R. Cranley and T. N. L. Patterson. Randomization of number theoretic methods for multiple integration. *SIAM Journal on Numerical Analysis*, 13(6):904–914, 1976.

C. de Boor. *A Practical Guide to Splines*. Springer-Verlag, New York, second edition, 2001.

Pierre Del Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006.

V. Demers, P. L'Ecuyer, and B. Tuffin. A combination of randomized quasi-Monte Carlo with splitting for rare-event simulation. In *Proceedings of the 2005 European Simulation and Modeling Conference*, pages 25–32, Ghent, Belgium, 2005. EUROSIS.

J. Dick and F. Pillichshammer. *Digital Nets and Sequences: Discrepancy Theory and Quasi-Monte Carlo Integration*. Cambridge University Press, Cambridge, U.K., 2010.

J. Dieudonné. *Foundations of Modern Analysis*. Academic Press, New York, second edition, 1969.

M. Dion and P. L'Ecuyer. American option pricing with randomized quasi-Monte Carlo simulations. In *Proceedings of the 2010 Winter Simulation Conference*, pages 2705–2720, 2010.

B. Efron and T. Hastie. *Computer Age Statistical Inference*. Cambridge University Press, New York, 2016.

R. El Haddad, C. Lécot, and P. L'Ecuyer. Quasi-Monte Carlo simulation of discrete-time Markov chains on multidimensional state spaces. In A. Keller, S. Heinrich, and H. Niederreiter, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2006*, pages 413–429, Berlin, 2008. Springer-Verlag.

R. El Haddad, C. Lécot, P. L'Ecuyer, and N. Nassif. Quasi-Monte Carlo methods for Markov chains with continuous multidimensional state space. *Mathematics and Computers in Simulation*, 81:560–567, 2010.

B. L. Fox and P. W. Glynn. Discrete-time conversion for simulating finite-horizon Markov processes. *SIAM Journal on Applied Mathematics*, 50:1457–1473, 1990.

M. Fu and J.-Q. Hu. *Conditional Monte Carlo*. Kluwer Academic, Boston, 1997.

M. C. Fu, D. B. Madan, and T. Wang. Pricing continuous Asian options: A comparison of Monte Carlo and Laplace transform inversion methods. *Journal of Computational Finance*, 2:49–74, 1998.

M. C. Fu, L. J. Hong, and J.-Q. Hu. Conditional Monte Carlo estimation of quantile sensitivities. *Management Science*, 55(12):2019–2027, 2009.

M. Gerber and N. Chopin. Sequential quasi-Monte Carlo. *Journal of the Royal Statistical Society, Series B*, 77(Part 3):509–579, 2015.

I. B. Gertsbakh and Y. Shpungin. *Models of Network Reliability*. CRC Press, Boca Raton, FL, 2010.

M. B. Giles. Multilevel Monte Carlo path simulation. *Operations Research*, 56(3):607–617, 2008.

M. B. Giles. Algorithm 955: approximation of the inverse Poisson cumulative distribution. *ACM Transactions on Mathematical Software*, 42:1–22, 2016.

D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361, 1977. doi: 10.1021/j100540a008.

D. T. Gillespie. The chemical Langevin equation. *The Journal of Chemical Physics*, 113(1): 297–306, 2000.

D. T. Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. *The Journal of Chemical Physics*, 115(4):1716–1733, 2001. doi: 10.1063/1.1378322.

P. Glasserman. *Monte Carlo Methods in Financial Engineering*. Springer-Verlag, New York, 2004.

P. W. Glynn. Likelihood ratio gradient estimation: an overview. In *Proceedings of the 1987 Winter Simulation Conference*, pages 366–375, Piscataway, NJ, 1987. IEEE Press.

W. B. Gong and Y.-C. Ho. Smoothed (conditional) perturbation analysis of discrete event dynamical systems. *IEEE Transactions on Automatic Control*, AC-32(10):858–866, 1987.

S. Haber. A modified Monte Carlo quadrature. *Mathematics of Computation*, 19:361–368, 1966.

M. Hardy. Combinatorics of partial derivatives. *Electron. J. Combin.*, 13(1):Research Paper 1, 13, 2006. ISSN 1077-8926. URL http://www.combinatorics.org/Volume_13/Abstracts/v13i1r1.html.

Z. He and X. Wang. On the convergence rate of randomized quasi-monte carlo for discontinuous functions. *SIAM Journal on Numerical Analysis*, 53(5):2488–2503, 2015.

A. Hellander. Efficient computation of transient solutions of the chemical master equation based on uniformization and quasi-Monte Carlo. *The Journal of Chemical Physics*, 128: 154109, 2008.

F. J. Hickernell. Obtaining $O(N^{-2+\epsilon})$ convergence for lattice quadrature rules. In K.-T. Fang, F. J. Hickernell, and H. Niederreiter, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2000*, pages 274–289, Berlin, 2002. Springer-Verlag.

D. J. Higham. Modeling and simulating chemical reactions. *SIAM Review*, 50(2):347–368, 2008. URL https://doi.org/10.1137/060666457.

N. J. Higham. The scaling and squaring method for the matrix exponential revisited. *SIAM Review*, 51(4):747–764, 2009.

H. S. Hong and F. H. Hickernell. Algorithm 823: Implementing scrambled digital sequences. *ACM Transactions on Mathematical Software*, 29:95–109, 2003.

J. Hong, Z. Hu, and G. Liu. Monte Carlo methods for value-at-risk and conditional value-at-risk: A review. *ACM Transactions on Modeling and Computer Simulation*, 24(4):Article 22, 2014.

S. Joe and F. Y. Kuo. Constructing Sobol sequences with better two-dimensional projections. *SIAM Journal on Scientific Computing*, 30(5):2635–2654, 2008.

M. C. Jones, J. S. Marron, and S. J. Sheather. A brief survey of bandwidth selection for density estimation. *Journal of the American Statistical Association*, 91(433):401–407, 1996.

P. E. Kloeden and E. Platen. *Numerical Solutions of Stochastic Differential Equations*. Springer-Verlag, Berlin, 1992.

W. Koh and K. T. Blackwell. Improved spatial direct method with gradient-based diffusion to retain full diffusive fluctuations. *The Journal of Chemical Physics*, 137(15):154111, 2012. doi: 10.1063/1.4758459. URL https://doi.org/10.1063/1.4758459.

P. J. Laub, R. Salomone, and Z. I. Botev. Monte Carlo estimation of the density of the sum of dependent random variables. *Mathematics and Computers in Simulation*, 161:23–31, 2019.

A. M. Law. *Simulation Modeling and Analysis*. McGraw-Hill, New York, fifth edition, 2014.

C. Lécot. Low discrepancy sequences for solving the Boltzmann equation. *Journal of Computational and Applied Mathematics*, 25(2):237–249, 1989.

C. Lécot and I. Coulibaly. A quasi-Monte Carlo scheme using nets for a linear Boltzmann equation. *SIAM Journal on Numerical Analysis*, 35(1):51–70, 1998.

C. Lécot and B. Tuffin. Quasi-Monte Carlo methods for estimating transient measures of discrete time Markov chains. In H. Niederreiter, editor, *Monte Carlo and Quasi-Monte Carlo Methods 2002*, pages 329–343, Berlin, 2004. Springer-Verlag.

P. L'Ecuyer. A unified view of the IPA, SF, and LR gradient estimation techniques. *Management Science*, 36(11):1364–1383, 1990.

P. L'Ecuyer. Quasi-Monte Carlo methods with applications in finance. *Finance and Stochastics*, 13(3):307–349, 2009.

P. L'Ecuyer. Random number generation. In J. E. Gentle, W. Haerdle, and Y. Mori, editors, *Handbook of Computational Statistics*, pages 35–71. Springer-Verlag, Berlin, second edition, 2012.

P. L'Ecuyer. SSJ: Stochastic simulation in Java. http://simul.iro.umontreal.ca/ssj/, 2016.

P. L'Ecuyer. Randomized quasi-Monte Carlo: An introduction for practitioners. In P. W. Glynn and A. B. Owen, editors, *Monte Carlo and Quasi-Monte Carlo Methods: MCQMC 2016*, pages 29–52, Berlin, 2018. Springer.

P. L'Ecuyer and E. Buist. Simulation in Java with SSJ. In M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, editors, *Proceedings of the 2005 Winter Simulation Conference*, pages 611–620, Piscataway, NJ, 2005. IEEE Press.

P. L'Ecuyer and E. Buist. On the interaction between stratification and control variates, with illustrations in a call center simulation. *Journal of Simulation*, 2(1):29–40, 2008.

P. L'Ecuyer and C. Lemieux. Quasi-Monte Carlo via linear shift-register sequences. In *Proceedings of the 1999 Winter Simulation Conference*, pages 632–639. IEEE Press, 1999.

P. L'Ecuyer and C. Lemieux. Variance reduction via lattice rules. *Management Science*, 46 (9):1214–1235, 2000.

P. L'Ecuyer and C. Lemieux. Recent advances in randomized quasi-Monte Carlo methods. In M. Dror, P. L'Ecuyer, and F. Szidarovszky, editors, *Modeling Uncertainty: An Examination of Stochastic Theory, Methods, and Applications*, pages 419–474. Kluwer Academic, Boston, 2002.

P. L'Ecuyer and D. Munger. On figures of merit for randomly-shifted lattice rules. In H. Woźniakowski and L. Plaskota, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2010*, pages 133–159, Berlin, 2012. Springer-Verlag.

P. L'Ecuyer and D. Munger. Algorithm 958: Lattice builder: A general software tool for constructing rank-1 lattice rules. *ACM Transactions on Mathematical Software*, 42(2): Article 15, 2016.

P. L'Ecuyer and G. Perron. On the convergence rates of IPA and FDC derivative estimators. *Operations Research*, 42(4):643–656, 1994.

P. L'Ecuyer and C. Sanvido. Coupling from the past with randomized quasi-Monte Carlo. *Mathematics and Computers in Simulation*, 81(3):476–489, 2010.

P. L'Ecuyer and R. Simard. Inverting the symmetrical beta distribution. *ACM Transactions on Mathematical Software*, 32(4):509–520, 2006.

P. L'Ecuyer, C. Lécot, and B. Tuffin. Randomized quasi-Monte Carlo simulation of Markov chains with an ordered state space. In H. Niederreiter and D. Talay, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2004*, pages 331–342, Berlin, 2006. Springer-Verlag.

P. L'Ecuyer, V. Demers, and B. Tuffin. Rare-events, splitting, and quasi-Monte Carlo. *ACM Transactions on Modeling and Computer Simulation*, 17(2):Article 9, 45 pages, 2007.

P. L'Ecuyer, C. Lécot, and B. Tuffin. A randomized quasi-Monte Carlo simulation method for Markov chains. *Operations Research*, 56(4):958–975, 2008.

P. L'Ecuyer, C. Lécot, and A. L'Archevêque-Gaudet. On array-RQMC for Markov chains: Mapping alternatives and convergence rates. In P. L'Ecuyer and A. B. Owen, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2008*, pages 485–500, Berlin, 2009. Springer-Verlag.

P. L'Ecuyer, D. Munger, C. Lécot, and B. Tuffin. Sorting methods and convergence rates for Array-RQMC: Some empirical comparisons. *Mathematics and Computers in Simulation*, 143:191–201, 2018.

P. L'Ecuyer, P. Marion, M. Godin, and F. Fuchhammer. A tool for custom construction of QMC and RQMC point sets. In *Monte Carlo and Quasi-Monte Carlo Methods: MCQMC 2020*, 2020a. submitted manuscript, available at http://www.iro.umontreal.ca/~lecuyer/myftp/papers/mcqmc20latnet.pdf.

P. L'Ecuyer, F. Puchhammer, and A. Ben Abdellah. Monte Carlo and quasi-Monte Carlo density estimation via conditioning. manuscript, arXiv:1906.04607v2, 2020b.

L. Lei, Y. Peng, M. C. Fu, and J.-Q. Hu. Applications of generalized likelihood ratio method to distribution sensitivities and steady-state simulation. *Discrete Event Dynamic Systems*, 28(1):109–125, 2018. ISSN 1573-7594.

C. Lemieux. *Monte Carlo and Quasi-Monte Carlo Sampling*. Springer-Verlag, 2009.

C. Lemieux and P. L'Ecuyer. Randomized polynomial lattice rules for multivariate integration and simulation. *SIAM Journal on Scientific Computing*, 24(5):1768–1789, 2003.

C. Lemieux, M. Cieslak, and K. Luttmer. *RandQMC User's Guide: A Package for Randomized Quasi-Monte Carlo Methods in C*, 2004. Software user's guide, available at http://www.math.uwaterloo.ca/~clemieux/randqmc.html.

D. B. Madan and E. Seneta. The variance gamma (V.G.) model for share market returns. *Journal of Business*, 63:511–524, 1990.

D. B. Madan, P. P. Carr, and E. C. Chang. The variance gamma process and option pricing. *European Finance Review*, 2:79–105, 1998.

P. Marion, M. Godin, and P. L'Ecuyer. An algorithm to compute the t-value of a digital net and of its projections. *Journal of Computational and Applied Mathematics*, 371, 2020. URL http://www.iro.umontreal.ca/~lecuyer/myftp/papers/tvalue.pdf.

J. S. Marron and M. P. Wand. Exact mean integrated square error. *The Annals of Statistics*, 20(2):712–736, 1992.

J. Matoušek. On the L_2-discrepancy for anchored boxes. *J. of Complexity*, 14:527–556, 1998.

M. K. Nakayama. Confidence intervals for quantiles using sectioning when applying variance-reduction techniques. *ACM Transactions on Modeling and Computer Simulation*, 24(4): Article 9, 2014a.

M. K. Nakayama. Quantile estimation when applying conditional monte carlo. In *2014 International Conference on Simulation and Modeling Methodologies, Technologies, and Applications (SIMULTECH)*, pages 280–285. IEEE, 2014b.

H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*, volume 63 of *SIAM CBMS-NSF Reg. Conf. Series in Applied Mathematics*. SIAM, 1992.

A. B. Owen. Randomly permuted (t,m,s)-nets and (t,s)-sequences. In H. Niederreiter and P. J.-S. Shiue, editors, *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, volume 106 of *Lecture Notes in Statistics*, pages 299–317. Springer-Verlag, 1995.

A. B. Owen. Scrambled net variance for integrals of smooth functions. *Annals of Statistics*, 25(4):1541–1562, 1997.

A. B. Owen. Monto Carlo extension of Quasi-Monte Carlo. In *Proceedings of the 30th Conference on Winter Simulation*, WSC '98, pages 571–578, Los Alamitos, CA, USA, 1998a. IEEE Computer Society Press.

A. B. Owen. Latin supercube sampling for very high-dimensional simulations. *ACM Transactions on Modeling and Computer Simulation*, 8(1):71–102, 1998b.

A. B. Owen. Scrambling Sobol and Niederreiter-Xing points. *Journal of Complexity*, 14: 466–489, 1998c.

A. B. Owen. Variance with alternative scramblings of digital nets. *ACM Transactions on Modeling and Computer Simulation*, 13(4):363–378, 2003.

A. B. Owen. A randomized Halton algorithm in R. Technical report, Stanford University, 2017. arXiv:1706.02808.

M. A. J. Padgett and S. Ilie. An adaptive tau-leaping method for stochastic simulations of reaction-diffusion systems. *AIP Advances*, 6(3):035217, 2016.

E. Parzen. On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

Y. Peng, M. C. Fu, P. W. Glynn, and J.-Q. Hu. On the asymptotic analysis of quantile sensivitiy estimation by Monte Carlo simulation. In *Proceedings of the 2017 Winter Simulation Conference*, pages 2336–2347, Piscataway, NJ, 2017. IEEE Press.

Y. Peng, M. C. Fu, J.-Q. Hu Hu, and B. Heidergott. A new unbiased stochastic derivative estimator for discontinuous sample performances with structural parameters. *Operations Research*, 66(2):487–499, 2018.

Y. Peng, M. C. Fu, B. Heidergott, and H. Lam. Maximum likelihood estimation by Monte Carlo simulation: Towards data-driven stochastic modeling. *Operations Research*, 2020. to appear.

D. S. G. Pollock. Smoothing with cubic splines. Technical report, University of London, Queen Mary and Westfield College, London, 1993.

F. Puchhammer, A. Ben Abdellah, and P. L'Ecuyer. Variance reduction with array-rqmc for tau-leaping simulation of stochastic biological and chemical reaction networks. manuscript, 2020.

V. C. Raykar and R. Duraiswami. Fast optimal bandwidth selection for kernel density estimation. In *Proceedings of the 2006 SIAM International Conference on Data Mining*, pages 524–528, 2006.

M. D. Schields and J. Zhang. The generalization of latin hypercube sampling. *Reliability Engineering and System Safety*, 148:96–108, 2016.

D. W. Scott. *Multivariate Density Estimation*. Wiley, second edition, 2015.

R. J. Serfling. *Approximation Theorems for Mathematical Statistics*. Wiley, New York, NY, 1980.

I. M. Sobol'. The distribution of points in a cube and the approximate evaluation of integrals. *U.S.S.R. Comput. Math. and Math. Phys.*, 7(4):86–112, 1967.

R. Strehl and S. Ilie. Hybrid stochastic simulation of reaction-diffusion systems with slow and fast dynamics. *The Journal of Chemical Physics*, 143(23):234108, 2015. doi: 10.1063/1.4937491. URL https://doi.org/10.1063/1.4937491.

G. R. Terrell and D. W. Scott. Variable kernel density estimation. *The Annals of Statistics*, 20(3):1236–1265, 1992.

A. W. Van der Vaart. *Asymptotic Statistics*. Cambridge University Press, 2000.

C. Wächter and A. Keller. Efficient simultaneous simulation of Markov chains. In A. Keller, S. Heinrich, and H. Niederreiter, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2006*, pages 669–684, Berlin, 2008. Springer-Verlag.

M. P. Wand and M. C. Jones. *Kernel Smoothing*. Chapman and Hall, 1995.

Appendix A

Supplement to Chapter 3

In this appendix, We examine empirically how RQMC estimators can achieve large IV and MISE reductions and even faster convergence rates than MC by using the histogram estimator and we give detailed results for the examples presented in Chapter 3.

A.1. A normalized sum of standard normals

Let $Z_1, \ldots, Z_s$ be s independent standard normal random variables generated by inversion and put $X = (a_1 Z_1 + \cdots + a_s Z_s)/\sigma$, where $\sigma^2 = a_1^2 + \cdots + a_s^2$. We estimate the density over the interval $(a,b) = (-b,b) = (-2,2)$. In our first experiment, we take $a_1 = \cdots = a_s = 1$, so all the coordinates have the same importance.

Table A.5 summarizes the results with histogram for $b = 2$, when B is estimated. For MC, our estimates given in the first column are based on experiments made with $s = 1$, but are valid for all s, because the IV and ISB do not depend on s. The estimated values for MC agree with the theory: the exact asymptotic values are $\gamma = 0.4$ and $\nu = 2/3$ and $\beta = \delta = 1$. The other columns give some results for Sobol'+LMS and Sobol'+NUS, for selected values of s. For all $s > 1$ that we have tried, LMS and NUS have a similar behavior. The first rows give the dimension s, the ℓ_0 found by pilot runs and used to fit the IV model, the estimated parameters C, β, and δ of the IV model, the fraction R^2 of variance explained by this model, and the estimated B. Recall that the rates $\tilde{\nu}$ and the e19 were obtained from a second-stage experiment, by using the estimated $\hat{h}_*(n)$ from the model in the first stage. All the R^2 coefficients are pretty close to 1, which means that the log-log linear model is reasonably good in the area considered. The estimate of B given in the tables turns out to be the same for all s and all RQMC methods, up to three decimal digits: it is $B \approx 0.01081$.

Thus, the estimator of B has very little variance. The MISE reduction of RQMC vs MC can be assessed by comparing their e19s given in the last row, for a given s. For example, for $s = 1$, the MISE for $n = 2^{19}$ is approximately 2^{-21} for Sobol'+NUS compared to 2^{-14} for MC, i.e., about 2^7 times smaller. For $s = 2$, for both LMS and NUS, the MISE is about $2^{-16.3}$, which is about 4 times smaller than for MC.

	MC	NUS	LMS	NUS	LMS	NUS	LMS	NUS	NUS	NUS	NUS
s		1	2	2	3	3	5	5	10	20	100
ℓ_0	5.0	8.5	6.0	6.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0
C	0.813	1.316	1.222	1.299	0.769	0.744	0.948	1.057	0.697	0.772	0.741
β	1.000	2.007	1.502	1.504	1.274	1.274	1.058	1.071	1.004	1.007	0.996
δ	1.041	2.012	2.022	2.012	1.893	1.906	1.181	1.196	1.094	1.077	1.051
R^2	1.000	1.000	1.000	1.000	0.995	0.995	0.998	0.998	1.000	1.000	1.000
B	0.011	0.011	0.011	0.011	0.011	0.011	0.011	0.011	0.011	0.011	0.011
$\hat{\kappa}_*$	3.339	3.315	3.249	3.304	2.949	2.918	3.459	3.571	3.162	3.275	3.237
$\hat{\gamma}_*$	0.329	0.500	0.373	0.375	0.327	0.326	0.333	0.335	0.324	0.327	0.326
ℓ_*	4.507	7.775	5.395	5.398	4.659	4.653	4.529	4.532	4.503	4.509	4.509
$\hat{K}_*$	0.352	0.237	0.227	0.235	0.193	0.189	0.348	0.368	0.306	0.331	0.329
$\hat{\nu}_*$	0.658	1.000	0.747	0.750	0.655	0.652	0.665	0.670	0.649	0.655	0.653
$\tilde{\nu}$	0.667	1.018	0.747	0.752	0.659	0.663	0.678	0.662	0.652	0.661	0.652
e19	13.98	21.10	16.29	16.29	14.64	14.66	14.08	14.07	14.00	13.99	13.99

Table A.1. Parameter estimates for the histogram estimator, for a sum of normals, over $(-2,2)$.

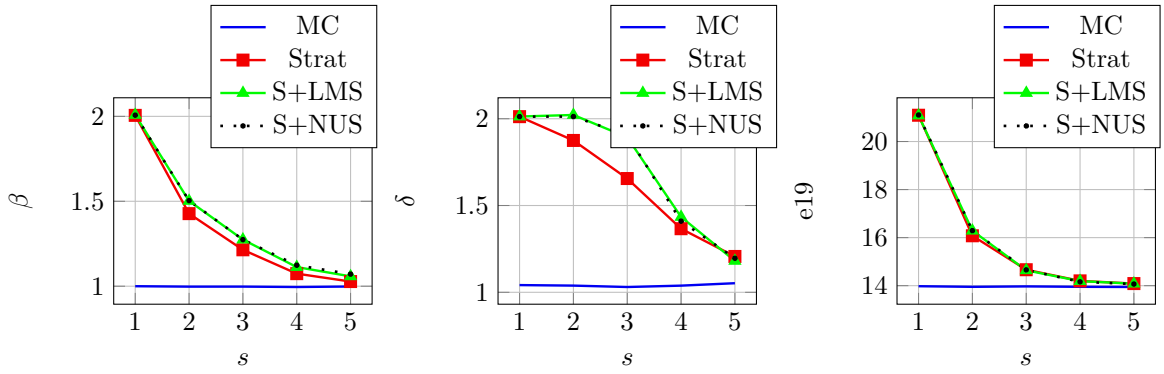

Figure A.1. Estimated β, δ, and e19 ($= -\log_2(\text{MISE})$ for $n = 2^{19}$) for the histogram over $(-2,2)$, with Monte Carlo, stratification, Sobol'+LMS, and Sobol'+NUS.

Figure A.1 shows the estimated β, δ, and e19 for the histogram density estimator, for $s = 1, \ldots, 5$. Stratification, shown in these plots and not in the tables, is exactly equivalent to Sobol'+NUS for $s = 1$, and somewhat less effective for $s > 1$.

One important observation from the plots and the last row of the tables (the e19) is that for all s, the RQMC methods never have a larger MISE than MC. Their MISE is much

smaller for very small s, and becomes almost the same as for MC when s gets large. In particular, for $b = 2$ we have $R(f') \approx 0.13456$ and $B = R(f')/12 \approx 0.01121$. The value of B obtained by estimating the derivative of the density is 0.0108. We repeated the density estimation experiment by using the exact values of B instead of the estimated ones to choose h, and the results were very close for all s. In particular, the MISE rates $\tilde{\nu}$ and the MISE values for $n = 2^{19}$ (the e19) were almost the same. For example, for MC with the exact B, the e19 was 13.91 (compared with 13.97 the estimated B). This is not surprising, because the values of B and h do not change much.

We now take decreasing coefficients in the linear combination, namely $a_j = 2^{-j}$ for $j = 1, \ldots, s$. Table A.2 summarizes our findings for Sobol'+LMS, for s up to 100. The results with Sobol'+NUS are very similar. For $s = 1$, the results are obviously the same as for our previous setting, but they diverge when we increase s. For example, in the previous setting, the MISE estimate with $n = 2^{19}$ for $s = 2$, 10, and 100, was $2^{-16.29}$, $2^{-14.00}$, and $2^{-13.99}$, respectively, whereas with the new weights, it is $2^{-16.47}$, $2^{-14.13}$, and $2^{-14.15}$, respectively. For all s, the MISE with RQMC and $n = 2^{19}$ was about the same as for MC in the previous setting and it is not improved, but the constant is improved (empirically). As expected, when s increases beyond about 10, all the model parameters appear to stabilize as a function of s. In the previous setting, they were stabilizing around the MC values, but now they stabilize to different values. For example, in $s = 100$ dimensions, β was near the MC value of 1, and now it is about 1.14.

s	MC	2	4	10	20	50	100
C	0.843	1.085	0.636	0.304	0.334	0.371	0.346
β	1.008	1.507	1.247	1.133	1.140	1.144	1.143
δ	1.037	2.024	2.077	1.891	1.887	1.857	1.883
$\hat{K}_*$	0.361	0.212	0.159	0.120	0.126	0.134	0.128
$\hat{\nu}_*$	0.664	0.749	0.612	0.582	0.587	0.593	0.589
$\tilde{\nu}$	0.661	0.751	0.627	0.613	0.602	0.610	0.617
e19	14.08	16.47	14.28	14.13	14.14	14.16	14.12

Table A.2. Parameter estimates for the Histogram under Sobol'+LMS, for a weighted sum of normals with $a_j = 2^{-j}$ over [−2,2].

A.2. Displacement of a cantilevel beam

Here, the exact density is unknown, so we will not be able to compute the ISB and the MISE, but we will estimate the AISB as we did in the previous example, and then use it to estimate the optimal h and the MISE. For the experiments reported here, we estimate the density over the interval (0.407,1.515), which covers about 99% of the density (it excludes roughly 0.5 % on each side). We also tried the shorter interval (0.590,1.293), which excludes 5% of the density on each side, and the results were very similar, except that the estimated constant B was about 17% smaller for the histogram.

Table A.9 gives the parameter estimates from our experiment, with the histogram estimator. RQMC increases the rate β significantly. For the KDE with Sobol' points, it goes from 1 to about 2. However, δ increases even more, from 1 to about 4. This means that although the variance decreases much faster than for MC as a function of n for fixed h, we cannot afford to decrease h very much to decrease the bias, so the MISE reduction is more limited than the IV reduction. The R^2 coefficient is very close to 1, showing that the linear model for $\log_2(\text{IV})$ fits very well. It is reassuring to see that the estimate of B is about the same for all point sets. The estimated convergence rate of the MISE, $\hat{\nu}_*$, is not improved by RQMC. However, RQMC reduces significantly the constant K in the MISE model.

Figure A.2 shows the estimated MISE as a function of n (with the estimated optimal h), as well as the estimated IV as a function of n, all in log scale, for the histogram. The results for Sobol'+LMS and Sobol'+NUS are practically indistinguishable in those plots. We can see that although the MISE rate (the slope) is not improved much by RQMC, the MISE is nevertheless reduced by a significant factor. For fixed h, the estimated IV converges at a faster rate with RQMC than with MC, as shown in the right part of the figure.

A.3. A weighted sum of lognormals

Here, We will estimate the density of X by histogram over the interval $[a,b] = [K, K + 27.13]$. Approximately 0.5% of the density lies on the right of this interval and 29.05% lies on the left (this is when the option brings no payoff).

Table A.11 summarizes the results of our experiments. Again, the linear model for the IV fits quite well in the selected area. With the histogram, RQMC improves β from 1 to about 1.14, but at the same time δ increases (unfortunately) from about 1.2 to nearly 2.13. This

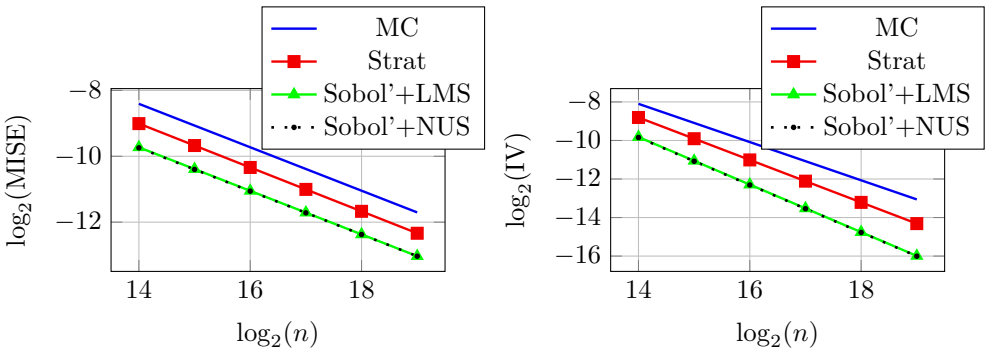

Figure A.2. Estimated MISE as a function of n for the cantilever example for the histogram (left) and Estimated IV as a function of n for fixed h; we took $h = 2^{-6} = 1/64$ (right).

	MC	Strat	LMS	NUS
C	0.831	0.424	0.130	0.119
β	0.992	1.102	1.234	1.232
δ	1.010	1.309	1.733	1.744
R^2	1.000	0.997	0.993	0.993
B	1.177	1.178	1.178	1.177
$\hat{\kappa}_*$	0.710	0.646	0.533	0.523
$\hat{\gamma}_*$	0.330	0.333	0.331	0.329
ℓ_*	6.758	6.962	7.188	7.186
$\hat{K}_*$	1.768	1.243	0.721	0.692
$\hat{\nu}_*$	0.659	0.666	0.661	0.658
e19	11.70	12.34	13.03	13.03

Table A.3. Experimental results for the density estimation of the displacement of a cantilever beam, with a histogram, over the interval (0.407,1.515).

	MC	LMS	NUS
C	0.765	0.398	0.429
β	1.015	1.140	1.146
δ	1.168	2.105	2.133
R^2	0.998	0.998	0.999
B	1.1E-5	1.1E-5	1.1E-5
$\hat{\kappa}_*$	28.26	12.97	13.03
$\hat{\gamma}_*$	0.320	0.278	0.277
ℓ_*	1.269	1.581	1.565
$\hat{K}_*$	0.024	0.004	0.004
$\hat{\nu}_*$	0.641	0.556	0.554
e19	17.53	18.63	18.61

Table A.4. Experimental results for the density estimation of the option payoff, with a histogram, over the interval (0,27.13).

means we are very limited in how much we can decrease h to reduce the bias. The estimate of B is about the same for all point sets, which is reassuring. Somewhat surprisingly, the

estimated MISE rate $\hat{\nu}_*$ is a bit worse for RQMC than for MC, due to the large δ. But the MISE is nevertheless significantly smaller for RQMC than for MC in the range of interest (about half), as shown in the left panel of Figure A.3, for which h was taken as the estimated optimal h from our model, as a function of n. That is, RQMC is truly beneficial for estimating the payoff density in this example. In the right panel, we see that the estimated IV for fixed h converges faster with RQMC than with MC.

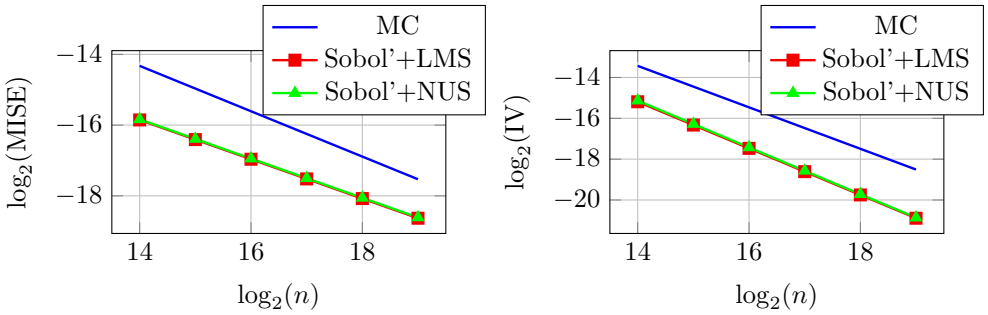

Figure A.3. Estimated MISE as a function of n for the option payoff example for the histogram (left) and estimated IV as a function of n for $h = 1/2$ for the histogram (right).

For comparison, when estimating the mean of X instead of the density, with Sobol'+LMS, the variance converges approximately as $\mathcal{O}(n^{-1.9})$ compared with $\mathcal{O}(n^{-1})$ for MC, and the variance is divided by a factor of about two millions compared with MC for $n = 2^{20}$. See L'Ecuyer (2018), Table 3.

A.4. Detailed Numerical Results

Here, we provide the detailed parameter estimates for the histogram and KDE estimators, for the examples described in Chapter 3.

		ℓ_0	C	β	δ	R^2	B	$\hat{\kappa}_*$	$\hat{\gamma}_*$	ℓ_*	$\hat{K}_*$	$\hat{\nu}_*$	$\tilde{\nu}$	e19
s=1	MC	5.0	0.813	1.000	1.041	1.000	0.015	2.972	0.329	4.677	9.109	0.657	0.669	13.98
	Strat	8.5	1.296	2.006	2.012	1.000	0.015	3.047	0.499	7.895	9.708	1.000	0.997	21.04
	LMS	8.5	1.330	2.008	2.013	1.000	0.015	3.066	0.500	7.895	9.836	1.001	0.994	21.02
	NUS	8.5	1.316	2.007	2.012	1.000	0.015	3.059	0.500	7.895	0.278	1.000	0.992	21.01
	Lat+s	8.5	0.522	1.992	2.009	1.000	0.015	2.640	0.497	8.040	0.149	0.994	1.004	21.63
	Lat+s+b	8.5	1.342	2.016	2.017	1.000	0.015	3.337	0.502	7.798	0.237	1.004	0.991	21.15
s=2	MC	5.0	0.786	0.997	1.038	1.000	0.015	2.942	0.328	4.680	8.920	0.656	0.664	13.94
	Strat	6.0	1.049	1.427	1.875	1.000	0.015	2.947	0.368	5.436	9.040	0.7367	0.738	16.05
	LMS	6.0	1.222	1.502	2.022	1.000	0.015	2.998	0.373	5.513	0.267	0.747	0.757	16.31
	NUS	6.0	1.299	1.504	2.012	1.000	0.015	3.049	0.375	5.513	0.276	0.750	0.755	16.29
	Lat+s	6.0	1.506	1.526	1.940	0.956	0.015	3.483	0.387	5.561	0.264	0.775	0.743	16.65
	Lat+s+b	6.0	1.694	1.515	2.021	0.994	0.015	3.533	0.377	5.338	0.266	0.754	0.773	16.23
s=3	MC	5.0	0.816	0.997	1.030	1.000	0.015	2.976	0.329	4.680	9.133	0.658	0.661	13.96
	Strat	5.0	0.818	1.214	1.655	0.998	0.015	2.836	0.332	4.812	8.333	0.664	0.659	14.63
	LMS	5.0	0.769	1.274	1.893	0.995	0.015	2.714	0.327	4.780	0.226	0.655	0.670	14.63
	NUS	5.0	0.744	1.274	1.906	0.994	0.015	2.687	0.326	4.772	0.221	0.652	0.681	14.67
	Lat+s	5.0	8.106	1.513	1.843	0.925	0.015	5.497	0.394	5.025	0.674	0.788	0.759	15.54
	Lat+s+b	5.0	3.474	1.291	1.547	0.955	0.015	4.749	0.364	4.665	0.553	0.728	0.674	14.68
s=5	MC	5.0	0.761	0.998	1.052	1.000	0.015	2.910	0.327	4.673	8.730	0.654	0.661	13.96
	Strat	5.0	0.612	1.027	1.206	0.999	0.015	2.701	0.320	4.666	7.525	0.641	0.653	14.12
	LMS	5.0	0.948	1.058	1.181	0.998	0.015	3.107	0.333	4.684	0.395	0.665	0.677	14.05
	NUS	5.0	1.056	1.071	1.196	0.998	0.015	3.209	0.335	4.684	0.418	0.670	0.654	14.05
	Lat+s	5.0	0.372	1.140	1.752	0.903	0.015	2.489	0.304	4.461	0.142	0.608	0.604	14.37
	Lat+s+b	5.0	0.330	0.997	1.273	0.987	0.015	2.482	0.305	4.476	0.169	0.609	0.589	14.14
s=10	MC	5.0	0.685	0.986	1.042	1.000	0.015	2.813	0.324	4.666	8.155	0.648	0.664	13.98
	Strat	5.0	0.741	1.006	1.062	1.000	0.015	3.238	0.328	4.873	0.326	0.657	0.662	14.76
	LMS	5.0	0.764	1.012	1.092	1.000	0.015	2.909	0.327	4.680	8.728	0.655	0.655	14.02
	NUS	5.0	0.696	1.004	1.094	1.000	0.015	2.822	0.324	4.666	0.346	0.649	0.651	14.00
	Lat+s	5.0	1.563	1.113	1.195	0.992	0.015	4.050	0.348	4.603	0.469	0.697	0.709	14.33
	Lat+s+b	5.0	0.636	1.006	1.108	0.999	0.015	3.079	0.324	4.526	0.284	0.647	0.666	14.11
s=20	MC	5.0	0.817	1.004	1.056	1.000	0.015	2.975	0.329	4.673	9.124	0.657	0.664	13.95
	LMS	5.0	0.706	0.995	1.065	0.999	0.015	2.837	0.325	4.666	8.297	0.649	0.655	13.97
	NUS	5.0	0.772	1.007	1.076	1.000	0.015	2.920	0.327	4.673	0.375	0.655	0.653	13.98
	Lat+s	5.0	0.739	1.005	1.062	0.997	0.015	3.243	0.328	4.541	0.324	0.657	0.655	14.10
	Lat+s+b	5.0	0.723	1.011	1.085	0.999	0.015	3.214	0.328	4.541	0.314	0.655	0.655	14.12
s=100	MC	5.0	0.742	0.998	1.040	0.999	0.015	3.253	0.328	4.535	0.331	0.656	0.659	14.07
	LMS	5.0	0.864	1.013	1.063	1.000	0.015	3.125	0.331	4.640	10.04	0.661	0.667	13.99
	NUS	5.0	0.741	0.996	1.051	1.000	0.015	2.974	0.326	4.629	9.095	0.653	0.648	13.97
	Lat+s	5.0	0.709	1.002	1.066	0.999	0.015	3.197	0.327	4.536	0.315	0.654	0.655	14.09
	Lat+s+b	5.0	0.692	1.002	1.067	0.999	0.015	3.173	0.327	4.539	0.310	0.653	0.651	14.10

Table A.5. Parameter estimates for the histogram estimator, for a sum of normals, over $(-2,2)$.

		ℓ_0	C	β	δ	R^2	B	$\hat{\kappa}_*$	$\hat{\gamma}_*$	ℓ_*	$\hat{K}_*$	$\hat{\nu}_*$	$\tilde{\nu}$	e19
s=1	MC	4.5	0.265	1.037	1.134	0.999	0.042	1.121	0.202	3.675	0.299	0.808	0.780	17.01
	Strat	8.5	0.280	3.003	3.002	1.000	0.042	1.259	0.429	7.828	2.736	1.715	1.715	34.53
	LMS	8.5	0.128	3.093	3.230	0.953	0.042	1.133	0.428	7.966	1.762	1.711	1.710	34.10
	NUS	8.5	0.032	2.791	3.004	0.999	0.042	0.925	0.398	7.682	0.071	1.594	1.595	34.06
s=2	MC	4.5	0.135	0.982	1.145	0.999	0.042	0.984	0.191	3.647	1.076	0.763	0.792	16.99
	Strat	6.0	0.150	1.674	2.331	1.000	0.042	1.124	0.264	4.853	1.731	1.058	1.064	22.52
	LMS	6.0	0.243	2.112	3.196	1.000	0.042	1.238	0.293	5.265	0.221	1.174	1.176	24.39
	NUS	6.0	0.212	2.101	3.196	1.000	0.042	1.215	0.292	5.265	0.205	1.168	1.169	24.38
	Lat+s	6.0	0.015	1.977	3.242	0.977	0.042	0.840	0.273	5.437	0.047	1.092	1.085	25.17
	Lat+s+b	6.0	3.492	2.303	3.193	0.991	0.042	1.793	0.320	5.240	0.974	1.280	1.314	24.37
s=3	MC	4.5	0.165	1.002	1.153	0.999	0.042	1.024	0.194	3.662	1.261	0.778	0.798	17.01
	Strat	5.0	0.121	1.380	2.147	0.999	0.042	1.075	0.224	4.166	1.447	0.898	0.899	19.61
	LMS	5.0	0.144	1.786	3.383	0.995	0.042	1.156	0.242	4.387	0.163	0.967	0.976	20.79
	NUS	5.0	0.180	1.798	3.357	0.995	0.042	1.191	0.244	4.392	0.184	0.978	0.975	20.80
	Lat+s	5.0	45.92	2.381	3.454	0.954	0.042	2.508	0.319	4.741	3.573	1.277	1.403	22.43
	Lat+s+b	5.0	0.407	1.725	3.151	0.959	0.042	1.329	0.241	4.173	0.297	0.965	0.965	20.09
s=5	MC	4.5	0.181	1.010	1.146	0.998	0.042	1.042	0.196	3.667	1.355	0.785	0.757	16.89
	Strat	4.5	0.102	1.149	1.777	0.996	0.042	1.014	0.199	3.766	1.157	0.795	0.801	17.71
	LMS	4.5	0.140	1.301	2.295	0.979	0.042	1.109	0.207	3.776	0.173	0.826	0.832	17.88
	NUS	4.5	0.096	1.270	2.303	0.978	0.042	1.045	0.201	3.764	0.137	0.806	0.806	17.79
	Lat+s	4.5	0.094	1.572	3.020	0.974	0.042	1.079	0.224	4.146	0.132	0.896	0.903	19.94
	Lat+s+b	4.5	0.014	1.055	2.090	0.981	0.042	0.754	0.173	3.699	0.040	0.693	0.713	17.83
s=10	MC	4.5	0.145	0.977	1.110	0.999	0.042	0.992	0.191	3.644	1.116	0.765	0.752	16.87
	Strat	4.0	0.108	1.040	1.464	0.996	0.042	0.986	0.190	3.829	0.151	0.762	0.749	17.96
	LMS	4.0	0.039	1.048	1.863	0.990	0.042	0.869	0.179	3.598	0.615	0.715	0.742	17.19
	NUS	4.0	0.029	1.011	1.811	0.990	0.042	0.820	0.174	3.591	0.061	0.696	0.744	17.28
	Lat+s	4.0	1.927	1.337	1.514	0.977	0.042	1.679	0.243	3.861	1.212	0.970	0.959	18.16
	Lat+s+b	4.0	0.041	1.025	1.783	0.987	0.042	0.865	0.177	3.578	0.076	0.709	0.737	17.19
s=20	MC	4.5	0.179	1.003	1.129	0.997	0.042	1.036	0.195	3.664	1.325	0.782	0.769	17.00
	LMS	4.0	0.069	1.001	1.471	0.993	0.042	0.913	0.183	3.608	0.772	0.732	0.771	17.10
	NUS	4.0	0.078	0.996	1.421	0.991	0.042	0.925	0.184	3.602	0.117	0.735	0.760	17.07
	Lat+s	4.0	0.080	1.008	1.448	0.991	0.042	0.935	0.185	3.614	0.121	0.740	0.765	17.12
	Lat+s+b	4.0	0.081	1.048	1.629	0.995	0.042	0.957	0.186	3.599	0.122	0.745	0.766	17.18
s=100	MC	4.5	0.164	0.997	1.138	0.998	0.042	1.021	0.194	3.655	0.206	0.776	0.777	17.02
	LMS	4.5	0.150	1.024	1.262	0.998	0.042	1.023	0.194	3.664	1.241	0.778	0.794	17.14
	NUS	4.5	0.077	0.967	1.264	0.997	0.042	0.901	0.184	3.642	0.745	0.735	0.773	17.03
	Lat+s	4.5	0.116	1.005	1.274	0.997	0.042	0.976	0.191	3.657	0.158	0.763	0.774	17.15
	Lat+s+b	4.5	0.123	1.023	1.332	0.996	0.042	0.995	0.192	3.650	0.165	0.767	0.756	17.17

Table A.6. Parameter estimates for the KDE, for a sum of normals, over $(-2,2)$.

		ℓ_0	C	β	δ	R^2	B	$\hat{\kappa}_*$	$\hat{\gamma}_*$	ℓ_*	$\hat{K}_*$	$\hat{\nu}_*$	$\tilde{\nu}$	e19
s=2	MC	4.0	0.843	1.008	1.037	0.999	0.011	3.392	0.332	4.544	0.361	0.664	0.661	14.08
	Strat	5.0	1.158	1.463	1.939	1.000	0.011	3.259	0.371	5.351	0.231	0.743	0.748	16.23
	LMS	5.0	1.085	1.507	2.024	1.000	0.011	3.161	0.375	5.457	0.212	0.749	0.751	16.47
	NUS	5.0	0.996	1.503	2.036	1.000	0.011	3.089	0.372	5.448	0.202	0.745	0.758	16.46
	Lat+s	5.0	1.014	1.562	2.064	0.993	0.011	3.089	0.384	5.674	0.201	0.769	0.788	16.92
	Lat+s+b	5.0	1.887	1.559	2.062	0.994	0.011	3.600	0.384	5.444	0.273	0.768	0.760	16.46
s=4	MC	4.0	0.759	1.002	1.053	0.999	0.011	3.273	0.328	4.525	0.333	0.656	0.653	14.06
	Strat	3.5	0.738	1.164	1.694	0.999	0.011	3.007	0.315	4.406	0.211	0.630	0.639	14.23
	LMS	3.5	0.636	1.247	2.077	0.996	0.011	2.749	0.306	4.353	0.159	0.612	0.627	14.28
	NUS	3.5	0.769	1.266	2.085	0.997	0.011	2.877	0.310	4.362	0.173	0.620	0.632	14.30
	Lat+s	3.5	0.034	1.045	2.039	0.911	0.011	1.341	0.259	4.492	0.038	0.517	0.559	14.55
	Lat+s+b	3.5	4.637	1.421	2.091	0.960	0.011	4.460	0.347	4.443	0.416	0.695	0.775	14.47
s=10	MC	4.0	0.722	0.997	1.052	0.999	0.011	3.221	0.327	4.519	0.322	0.653	0.665	14.05
	Strat	4.0	0.736	1.019	1.130	0.999	0.011	3.212	0.326	4.827	0.308	0.651	0.653	14.72
	LMS	4.0	0.304	1.133	1.891	0.991	0.011	2.331	0.291	4.311	0.120	0.582	0.613	14.13
	NUS	4.0	0.343	1.141	1.876	0.992	0.011	2.407	0.294	4.324	0.128	0.589	0.614	14.15
	Lat+s	4.0	0.053	1.042	2.024	0.964	0.011	1.490	0.259	4.345	0.053	0.518	0.539	14.24
	Lat+s+b	4.0	1.024	1.212	1.776	0.979	0.011	3.243	0.321	4.400	0.239	0.642	0.713	14.26
s=20	MC	4.0	0.777	1.001	1.042	0.999	0.011	3.301	0.329	4.531	0.340	0.658	0.662	14.06
	LMS	4.0	0.334	1.140	1.887	0.991	0.011	2.388	0.293	4.317	0.126	0.587	0.602	14.14
	NUS	4.0	0.347	1.143	1.879	0.992	0.011	2.413	0.295	4.328	0.129	0.129	0.601	14.16
	Lat+s	4.0	0.059	1.049	2.009	0.964	0.011	1.532	0.262	4.356	0.050	0.523	0.545	14.26
	Lat+s+b	4.0	0.932	1.207	1.795	0.975	0.011	3.155	0.318	4.385	0.225	0.636	0.691	14.24
s=50	MC	4.0	0.707	0.995	1.046	0.999	0.011	3.200	0.327	4.530	0.319	0.653	0.658	14.07
	LMS	4.0	0.371	1.144	1.857	0.990	0.011	2.460	0.296	4.334	0.134	0.593	0.610	14.16
	NUS	4.0	0.278	1.128	1.911	0.991	0.011	2.273	0.289	4.298	0.113	0.577	0.609	14.12
	Lat+s	4.0	0.055	1.040	1.999	0.964	0.011	1.504	0.260	4.351	0.048	0.520	0.535	14.25
	Lat+s+b	4.0	1.080	1.217	1.780	0.977	0.011	3.287	0.322	4.400	0.245	0.644	0.700	14.26
s=100	MC	4.0	0.751	0.999	1.040	0.999	0.011	3.264	0.328	4.535	0.333	0.657	0.662	14.07
	LMS	4.0	0.346	1.143	1.883	0.991	0.011	2.411	0.294	4.322	0.128	0.589	0.617	14.15
	NUS	4.0	0.296	1.133	1.902	0.992	0.011	2.312	0.290	4.307	0.117	0.581	0.607	14.12
	Lat+s	4.0	0.053	1.038	2.002	0.962	0.011	1.494	0.259	4.348	0.048	0.519	0.548	14.24
	Lat+s+b	4.0	0.891	1.198	1.770	0.976	0.011	3.129	0.318	4.394	0.223	0.636	0.686	14.24

Table A.7. Parameter estimates for the histogram, for a weighted sum of normals with $a_j = 2^{-j}$.

		ℓ_0	C	β	δ	R^2	B	$\hat{\kappa}_*$	$\hat{\gamma}_*$	ℓ_*	$\hat{K}_*$	$\hat{\nu}_*$	$\tilde{\nu}$	e19
s=2	MC	4.0	0.162	1.004	1.185	0.998	0.042	1.025	0.194	3.643	0.204	0.774	0.765	17.01
	Strat	6.0	0.147	1.708	2.410	1.000	0.042	1.124	0.266	4.893	0.177	1.066	1.063	22.75
	LMS	6.0	0.173	2.100	3.189	1.000	0.042	1.181	0.292	5.310	0.183	1.168	1.176	24.65
	NUS	6.0	0.173	2.102	3.198	1.000	0.042	1.182	0.292	5.307	0.183	1.168	1.181	24.64
	Lat+s	6.0	0.012	1.980	3.163	0.997	0.042	0.809	0.276	5.558	0.041	1.106	1.118	25.63
	Lat+s+b	6.0	0.125	2.092	3.282	0.998	0.042	1.131	0.287	5.281	0.152	1.149	1.150	24.55
s=4	MC	4.0	0.147	0.999	1.200	0.998	0.042	1.009	0.192	3.635	0.190	0.768	0.775	16.99
	Strat	4.5	0.083	1.254	2.103	0.997	0.042	1.007	0.206	3.898	0.125	0.822	0.833	18.64
	LMS	4.5	0.038	1.650	3.745	0.985	0.042	0.980	0.213	4.077	0.079	0.852	0.892	19.85
	NUS	4.5	0.046	1.671	3.762	0.983	0.042	1.006	0.215	4.082	0.088	0.861	0.878	19.87
	Lat+s	4.5	0.001	1.403	3.601	0.926	0.042	0.610	0.185	4.219	0.012	0.738	0.582	20.37
	Lat+s+b	4.5	0.461	1.821	3.650	0.959	0.042	1.352	0.238	4.088	0.293	0.952	1.043	19.86
s=10	MC	4.0	0.149	0.999	1.199	0.998	0.042	1.011	0.192	3.635	0.191	0.769	0.785	17.00
	LMS	4.5	0.007	1.420	3.626	0.975	0.042	0.776	0.186	3.903	0.032	0.745	0.750	19.13
	NUS	4.5	0.007	1.426	3.610	0.975	0.042	0.784	0.187	3.911	0.033	0.750	0.759	19.16
	Lat+s	4.5	0.002	1.303	3.335	0.954	0.042	0.631	0.178	4.039	0.015	0.710	0.606	19.60
	Lat+s+b	4.5	0.049	1.521	3.398	0.964	0.042	0.999	0.206	3.907	0.091	0.822	0.849	19.09
s=20	MC	4.0	0.144	0.999	1.216	0.998	0.042	1.008	0.192	3.628	0.186	0.766	0.777	16.98
	LMS	4.5	0.008	1.427	3.582	0.977	0.042	0.793	0.188	3.912	0.035	0.753	0.730	19.15
	NUS	4.5	0.007	1.424	3.608	0.976	0.041	70.783	0.187	3.909	0.033	0.749	0.747	19.14
	Lat+s	4.5	0.002	1.311	3.353	0.955	0.042	0.637	0.178	4.037	0.015	0.713	0.607	19.59
	Lat+s+b	4.5	0.046	1.515	3.391	0.961	0.042	0.992	0.205	3.907	0.088	0.820	0.840	19.08
s=50	MC	4.0	0.138	0.997	1.212	0.998	0.042	0.999	0.191	3.634	0.180	0.765	0.782	17.01
	LMS	4.5	0.007	1.425	3.604	0.977	0.042	0.784	0.187	3.912	0.033	0.750	0.758	19.16
	NUS	4.5	0.007	1.415	3.609	0.974	0.042	0.774	0.186	3.903	0.032	0.744	0.747	19.12
	Lat+s	4.5	0.002	1.308	3.360	0.954	0.042	0.633	0.178	4.037	0.015	0.711	0.613	19.59
	Lat+s+b	4.5	0.045	1.518	3.418	0.965	0.042	0.989	0.205	3.905	0.087	0.819	0.829	19.08
s=100	MC	4.0	0.155	1.004	1.196	0.998	0.042	1.019	0.193	3.642	0.197	0.773	0.785	17.02
	LMS	4.5	0.008	1.429	3.603	0.975	0.042	0.792	0.188	3.907	0.035	0.752	0.752	19.14
	NUS	4.5	0.007	1.425	3.626	0.976	0.042	0.781	0.187	3.906	0.033	0.747	0.750	19.14
	Lat+s	4.5	0.002	1.314	3.362	0.953	0.042	0.638	0.178	4.039	0.015	0.714	0.614	19.60
	Lat+s+b	4.5	0.178	1.003	1.130	0.999	0.042	1.036	0.195	3.662	0.219	0.782	0.785	17.04

Table A.8. Parameter estimates for the KDE, for a weighted sum of normals with $a_j = 2^{-j}$.

		C	β	δ	R^2	B	$\hat{\kappa}_*$	$\hat{\gamma}_*$	ℓ_*	$\hat{K}_*$	$\hat{\nu}_*$	e19
Histogram (α=2)	MC	0.831	0.992	1.010	1.000	1.177	0.710	0.330	6.758	1.768	0.659	11.70
	Strat	0.424	1.102	1.309	0.997	1.178	0.646	0.333	6.962	1.243	0.666	12.34
	LMS	0.130	1.234	1.733	0.993	1.178	0.533	0.331	7.188	0.721	0.661	13.03
	NUS	0.119	1.232	1.744	0.993	1.177	0.523	0.329	7.186	0.692	0.658	13.03
	Lat+s	0.091	1.250	1.768	0.969	1.181	0.489	0.331	7.333	0.604	0.663	13.33
	Lat+s+b	1.380	1.142	1.645	0.990	1.181	0.437	0.313	7.148	0.499	0.627	12.91
KDE (α=4)	Lat+s	0.054	1.315	2.032	0.979	1.181	0.467	0.326	7.298	0.510	0.652	13.37
	Lat+s+b	0.033	1.229	2.004	0.990	1.181	0.409	0.307	7.121	0.396	0.614	13.00

Table A.9. Experimental results for the density estimation of the displacement of a cantilever beam, over the interval [0.407,1.515].

		C	β	δ	R^2	B	$\hat{\kappa}_*$	$\hat{\gamma}_*$	ℓ_*	$\hat{K}_*$	$\hat{\nu}_*$	e19
Histogram (α=2)	MC	0.765	1.015	1.168	0.998	1.1E-5	28.26	0.320	1.269	0.024	0.641	17.53
	LMS	0.398	1.140	2.105	0.998	1.1E-5	12.97	0.278	1.581	0.004	0.556	18.63
	NUS	0.429	1.146	2.133	0.999	1.1E-5	13.03	0.277	1.565	0.004	0.554	18.61
	Lat+s	0.345	0.998	1.435	0.961	1.2E-5	17.94	0.291	1.357	0.009	0.581	17.78
	Lat+s+b	0.523	1.031	1.399	0.991	1.2E-5	20.75	0.303	1.385	0.013	0.606	17.81
KDE (α=4)	MC	0.171	1.005	1.151	0.999	1.1E-6	7.953	0.195	0.715	0.020	0.780	20.45
	LMS	0.110	1.671	4.907	0.990	1.1E-6	3.717	0.188	1.670	4E-4	0.750	25.59
	NUS	0.097	1.663	4.930	0.990	1.1E-6	3.657	0.186	1.668	4E-4	0.745	25.58
	Lat+s	0.102	0.990	1.341	0.965	1.2E-6	6.837	0.185	0.750	0.010	0.742	20.68
	Lat+s+b	0.167	1.020	1.240	0.997	1.2E-6	7.677	0.195	0.759	0.017	0.779	20.63

Table A.10. Experimental results for the density estimation of the option payoff over the interval (0,27.13) with lattice rule.

		C	β	δ	R^2	B	$\hat{\kappa}_*$	$\hat{\gamma}_*$	ℓ_*	$\hat{K}_*$	$\hat{\nu}_*$	e19
Histogram (α=2)	MC	0.765	1.015	1.168	0.998	1.1E-5	28.26	0.320	1.269	0.024	0.641	17.53
	LMS	0.398	1.140	2.105	0.998	1.1E-5	12.97	0.278	1.581	0.004	0.556	18.63
	NUS	0.429	1.146	2.133	0.999	1.1E-5	13.03	0.277	1.565	0.004	0.554	18.61
	Lat+s	0.345	0.998	1.435	0.961	1.2E-5	17.94	0.291	1.357	0.009	0.581	17.78
	Lat+s+b	0.523	1.031	1.399	0.991	1.2E-5	20.75	0.303	1.385	0.013	0.606	17.81
KDE (α=4)	Lat+s	0.102	0.990	1.341	0.965	1.2E-6	6.837	0.185	0.750	0.010	0.742	20.68
	Lat+s+b	0.167	1.020	1.240	0.997	1.2E-6	7.677	0.195	0.759	0.017	0.779	20.63

Table A.11. Experimental results for the density estimation of the option payoff over the interval (0,27.13) with lattice rule.

Appendix B

Supplement to Chapter 5

In this appendix, we provide additional experiments, by using classical RQMC method, for the examples presented in Chapter 5, to compare them with array-RQMC method.

B.1. Variance Gamma model

For comparison with array-RQMC, we made an experiment using classical RQMC with these methods for the same numerical example as provided here, but with $c = 8$ and $c = 16$ instead of $c = 10$, to have powers of 2, and $t_j = jT/c$. Table B.1 reports the estimated regression slopes $\hat{\beta}$ for $\log_2 \mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$ vs $\log_2(n)$, and VRF compared with MC for $n = 2^{20}$, denoted VRF20 for BGSS, BGBS, and DGBS. For all the RQMC points set, the VRF20 values in Table 5.1 are much larger that these in Table B.1, showing that Array-RQMC can provide much larger variance reductions than the classical RQMC method.

	BGSS		BGBS		DGBS	
Point sets	$\hat{\beta}$	VRF20	$\hat{\beta}$	VRF20	$\hat{\beta}$	VRF20
MC	-1.00 -1.00	1 1	-1.00 -1.00	1 1	-1.00 -1.00	1 1
Sob+LMS	-1.18 -1.36	183 85	-1.23 -1.31	1258 895	-1.46 -0.87	3405 550
Sob+NUS	-1.33 -1.26	201 84	-1.35 -1.31	1402 1122	-1.41 -0.79	5257 599
Lat+s+b	-1.52 -1.55	231 56	-1.47 -1.38	2626 1330	-1.24 -1.07	6535 1340

Table B.1. Regression slopes $\hat{\beta}$ and VRF20, c=8 (left) and c=16 (right), for BGSS , BGBS, and DGBS, for the Asian option under a variance gamma process.

B.2. Heston volatility model

We also made an experiment using only the classical RQMC for the same numerical example as given here. Table B.2 reports the estimated regression slopes $\hat{\beta}$ for $\log_2 \mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$

vs $\log_2(n)$, and VRF compared with MC for $n = 2^{20}$, denoted VRF20. For all the RQMC points set, the VRF20 values in Table 5.2 are much larger than those in Table B.2 , which shown that Array-RQMC can provide much larger variance reductions than the classical RQMC method.

	European		Asian	
Point sets	$\hat{\beta}$	VRF20	$\hat{\beta}$	VRF20
MC	-1.00	1	-1.00	1
Sob+LMS	-1.24	128	-1.25	173
Sob+NUS	-1.08	97	-1.13	194
Lat+s+b	-1.32	56	-1.38	50

Table B.2. Regression slopes $\hat{\beta}$ and VRF20, for the Asian option under the Heston model by using the RQMC method.

B.3. Ornstein-Uhlenbeck volatility model

We also made an experiment using only the classical RQMC for the same numerical example as given here. Table B.3 reports the estimated regression slopes $\hat{\beta}$ for $\log_2 \mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$ vs $\log_2(n)$, and VRF compared with MC for $n = 2^{20}$, denoted VRF20. For all the RQMC points set, the VRF20 values in Table 5.3 are much larger that these in Table B.3, which shows that Array-RQMC can provide much larger variance reductions than the classical RQMC method.

	European		Asian	
Point sets	$\hat{\beta}$	VRF20	$\hat{\beta}$	VRF20
MC	-1.00	1	-1.00	1
Sob+LMS	-1.20	23739	-1.03	16233
Sob+NUS	-1.10	24597	-1.08	15702
Lat+s+b	-0.96	5135	-0.96	4463

Table B.3. Regression slopes $\hat{\beta}$ and VRF20, for the Asian option under the Ornstein model.

Appendix C

Supplement to chapter 6

In this appendix, we provide additional experiments on two examples:
(1) Linear Birth-Death Process ($\ell = 1$, $d = 2$),
(2) Enzyme kinetic reaction ($\ell = 4$, $d = 3$),
(3) Mitogen activated protein kinase (MAPK) cascade model ($\ell = 11$, $d = 14$).

C.1. Linear Birth-Death Process

In this example we consider one single substance S_1 and two reaction channels with constant reaction rate c.

$$S_1 \xrightarrow{c} \emptyset$$

$$S_1 \xrightarrow{c} 2S_1.$$

This models auto-catalytic production and degradation of S_1. Here, the stoichiometric matrix is defined by $\boldsymbol{\zeta} = \{-1,1\}$ and the propensity functions are $a_1(x) = cx$ and $a_2(x) = cx$. For the mean and the variance of the number of molecules at step t, $\mathbf{X}_t$, it is known that $\mathbb{E}(\mathbf{X}_t) = \mathbf{x}_0$ and $\mathrm{Var}(\mathbf{X}_t) = 2ct\mathbf{x}_0$.

For our experiments we take $c = 1$, $\mathbf{x}_0 = 1000$, $\tau = 1$, $d = 2$ and $T = 8$.

Table C.1 reports the VRF as well as the regression slopes of $\log_2 \mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$ as a function of $\log_2 n$, estimated from $m = 100$ independent replications with linear regression using all values of n considered. We can clearly see that Array-RQMC applied to τ-leap method gives a strong improvement compared to the standard pseudo random numbers for all the point set used.

Sample	$\hat{\beta}$	VRF20
MC	1.04	1
Sob+LMS	1.40	32094
Sob+NUS	1.40	40464
Lat+s	1.73	27158
Lat+s+b	1.25	13415

Table C.1. Regression slopes for $\log_2(n)$ $\mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$ vs $\log_2 n$, VFR for RQMC vs MC for $m = 100$, $n = 2^{19}$

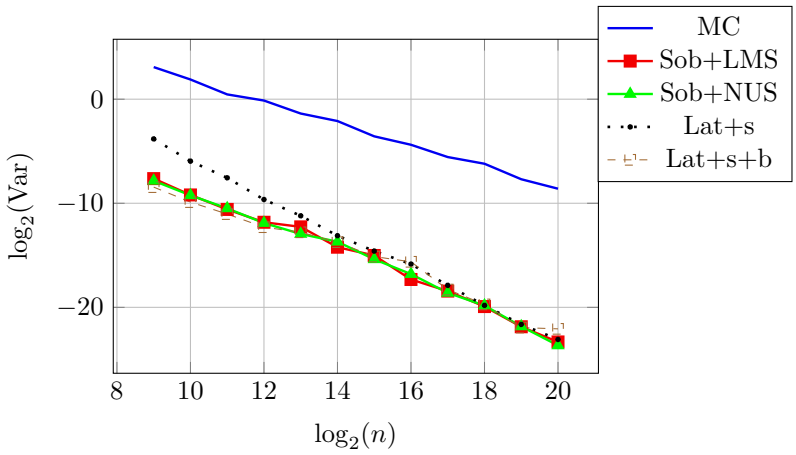

Figure C.1. Estimated Variance as a function of n for the Linear birth-deat hprocess.

Figure C.1 shows our estimate of $\mathrm{Var}[\hat{\mu}_n^{\mathrm{arqmc}}]$ as a function of $\log_2(n)$. The slope indicates a convergence rate of approximately $\mathrm{O}(n^{-3/2})$, and $\mathrm{O}(n^{-2})$ for Sobol and lattice, respectively.

C.2. Enzyme kinetic reaction

This example was taken from Ault (1974). In this model, an enzyme E is used to catalyze a substrate S into a final product P. This happens via an intermediary reversible reaction resulting in a fourth chemical species ES. This system can be explained graphically as follows

$$E + S \xrightarrow{c_1} ES,$$

$$ES \xrightarrow{c_2} E + S$$

$$ES \xrightarrow{c_3} E + P.$$

We fix the rates as $c_1 = 10^{-3}$, $c_2 = 10^{-4}$, $c_3 = 0.1$, as well as the initial values $\mathbf{x}_0 = (200, 500, 200, 0)$, and simulate this system for a duration of $T = 10$ using 7 steps. The state

dimension is $\ell = 4$ and the number of reactions is $d = 3$. Consequently, the dimensionality of this problem is either 4 or 7, depending on the sorting algorithm we use, as opposed to 21 with the traditional approach. We chose to investigate this model, as its number of reactions is slightly smaller than for the previous example, while the dimension of its state space is slightly higher.

At first, we chose the product E as the observed variable. We summarize our findings in Table C.2. For Sob+NUS, the sorting method which showed the smallest variance was the split sort, yielding a VRF20 of 3165 and a convergence rate of the variance of $n^{-1.33}$, although the other sort performed comparably well, i.e. VRF20 = 2523 for the batch sort with a convergence rate of $n^{-1.32}$.

Secondly, we chose the final product P as the observed variable. We summarize our findings in Table C.3. For Sob+NUS, the sorting method which showed the smallest variance was the split sort, yielding a VRF20 of 4041 and a convergence rate of the variance of $n^{-1.41}$, although the other sorts performed comparably well, i.e. VRF20 = 2939 for the batch sort with a convergence rate of $n^{-1.32}$ and VRF20 = 1493 for the Hilbert batch sort with a convergence rate of $n^{-1.09}$.

Sort	Sample	$\hat{\beta}$	VRF20
	MC	1.00	1
Split	Sob+LMS	1.30	2936
	Sob+NUS	1.33	3165
	Lat+s	1.59	5299
	Lat+s+b	1.33	3279
Batch	Sob+LMS	1.20	2387
	Sob+NUS	1.31	2523
	Lat+s	1.47	4615
	Lat+s+b	1.33	4538
Hilbert Batch	Sob+LMS	0.84	597
	Sob+NUS	0.80	449
	Lat+s	1.41	2319
	Lat+s+b	1.29	1863

Table C.2. Enzyme kinetic reaction, E : Estimated variance rates $\hat{\beta}$ and VRF20 with $m = 100$ and $n = 2^{20}$.

C.3. Mitogen activated protein kinase cascade model

We study the mitogen activated protein kinase (MAPK) cascade model as described in Padgett and Ilie (2016). Here, the transcription factor MAPK is phosphorylated in two

Sort	Sample	$\hat{\beta}$	VRF20
	MC	1.00	1
Split	Sob+LMS	1.40	3525
	Sob+NUS	1.41	4041
	Lat+s	1.59	5299
	Lat+s+b	1.33	3279
Batch	Sob+LMS	1.21	2749
	Sob+NUS	1.32	2939
	Lat+s	1.47	4615
	Lat+s+b	1.33	4538
Hilbert batch	Sob+LMS	1.16	1773
	Sob+NUS	1.09	1493
	Lat+s	1.41	2319
	Lat+s+b	1.29	1863

Table C.3. Enzyme kinetic reaction, P: Estimated variance rates $\hat{\beta}$ and VRF20 with $m = 100$ and $n = 2^{20}$.

steps by the upstream kinase MAPKK and dephosphorylated by the phosphatase P. The latter two species reach an inactive states MAPKK^* and P^*, respectively, after they released their product. Their re-activation occurs according to a certain time-scale $\tau_{\text{rel}} = 0.1$. The reactions are summarized below

$$\begin{aligned}
\text{MAPK} + \text{MAPKK} &\underset{c_2}{\overset{c_1}{\rightleftharpoons}} \text{MAPK_MAPKK},\\
\text{MAPK_MAPKK} &\xrightarrow{c_3} \text{MAPKK}^* + \text{MAPK}_{\text{P}},\\
\text{MAPKK}^* &\xrightarrow{c_4} \text{MAPKK},\\
\text{MAPK}_{\text{P}} + \text{MAPKK} &\underset{c_6}{\overset{c_5}{\rightleftharpoons}} \text{MAPK}_{\text{P}}\text{_MAPKK},\\
\text{MAPK}_{\text{P}}\text{_MAPKK} &\xrightarrow{c_7} \text{MAPKK} + \text{MAPK}_{\text{PP}},\\
\text{MAPK}_{\text{PP}} + \text{P}^* &\underset{c_2}{\overset{c_1}{\rightleftharpoons}} \text{MAPK}_{\text{PP}}\text{_P},\\
\text{MAPK}_{\text{PP}}\text{_P} &\xrightarrow{c_3} \text{P}^* + \text{MAPK}_{\text{P}},\\
\text{P}^* &\xrightarrow{c_4} \text{P},\\
\text{MAPK}_{\text{P}} + \text{P} &\underset{c_6}{\overset{c_5}{\rightleftharpoons}} \text{MAPK}_{\text{P}}\text{_P},\\
\text{MAPK}_{\text{P}}\text{_P} &\xrightarrow{c_7} \text{P} + \text{MAPK},
\end{aligned}$$

The rate constants are fixed as $c_1 = 0.027$, $c_2 = 1.35$, $c_2 = 1.35$, $c_3 = 1.5$, $c_4 = \log(2)/\tau_{\text{rel}}$, $c_5 = 0.028$, $c_6 = 1.73$, and $c_7 = 15$. Furthermore, we investigate these reactions up to the final time $T = 0.01$ and discretize with the step size $\tau = T/10 = 0.001$. This means, we have $\ell = 11$ and $d = 14$ and, thus, the traditional τ-leaping algorithm is 140-dimensional, while its dimensionality is either 15 or 25 in combination with array-RQMC.

The initial number of molecules are 2500 for MAPK and MAPK_{PP}, 8000 for MAPK_MAPKK, 7000 for MAPK_{P} and MAPK_{P}_P, 5500 for P^*, and 4500 for all other species.

The first species we observed was MAPK. Here, the performance of all sorts with Sobol+NUS was roughly the same. The smallest variance was obtained for the split sort (VRF20 = 2254) followed by the batch sort (VRF20 = 1314) and the hilbert batch sort (VRF20 = 1249). The convergence rate of the variance was always close to n^{-1}. The results are summarized in Table C.4

For MAPK_{PP} with Sobol+NUS. Here, the batch and Hilbert batch sort yielded the smallest variance (VRF20 = 1627), closely followed by the split sort (VRF20 = 1503). The results are summarized in Table C.5.

For MAPK_{PP}_P with Sobol+NUS, The split sort attained VRF20 = 1981, while we obtained VRF20 = 1465 for the batch sort and VRF20 = 1349 for the hilbert batch sort . The results are summarized in Table C.6.

Sort	Sample	$\hat{\beta}$	VRF20
	MC	1.00	1
Split	Sob+LMS	0.96	1394
	Sob+NUS	1.04	1656
	Lat+s	0.99	1990
	Lat+s+b	1.02	2055
Batch	Sob+LMS	0.93	1738
	Sob+NUS	0.87	1314
	Lat+s	1.13	2333
	Lat+s+b	0.98	1676
Hilbert batch	Sob+LMS	0.81	1132
	Sob+NUS	1.01	1249
	Lat+s	1.14	1729
	Lat+s+b	1.17	1716

Table C.4. MAPK cascade model, MAPK: Estimated variance rates $\hat{\beta}$ for $n = 2^{13}, \ldots, 2^{20}$ and VRF20 with $m = 100$.

Sort	Sample	$\hat{\beta}$	VRF20
	MC	1.00	1
Split	Sob+LMS	0.93	1227
	Sob+NUS	0.93	1503
	Lat+s	1.09	1829
	Lat+s+b	1.11	2455
Batch	Sob+LMS	1.09	2337
	Sob+NUS	0.98	1627
	Lat+s	1.08	1930
	Lat+s+b	1.01	1585
Hilbert batch	Sob+LMS	1.05	1377
	Sob+NUS	1.02	1626
	Lat+s	0.93	1218
	Lat+s+b	1.02	1487

Table C.5. MAPK cascade model, MAPKpp: Estimated variance rates $\hat{\beta}$ for $n = 2^{13}, \ldots, 2^{20}$ and VRF20 with $m = 100$..

Sort	Point sets	$\hat{\beta}$	VRF20
	MC	1.00	1
Split	Sob+LMS	1.01	1330
	Sob+NUS	1.06	1981
	Lat+s	1.22	2301
	Lat+s+b	0.99	1627
Batch	Sob+LMS	1.06	1753
	Sob+NUS	1.08	1465
	Lat+s	1.11	1501
	Lat+s+b	1.06	1531
Hilbert batch	Sob+LMS	1.04	1179
	Sob+NUS	1.02	1349
	Lat+s	1.07	1158
	Lat+s+b	0.95	1404

Table C.6. MAPK cascade model, $\text{MAPK}_{\text{PP\_P}}$: Estimated variance rates $\hat{\beta}$ for $n = 2^{13}, \ldots, 2^{20}$ and VRF20 with $m = 100$.