Zdravko Botev
Alexander Keller
Christiane Lemieux
Bruno Tuffin *Editors*

Advances in Modeling and Simulation

Festschrift for Pierre L'Ecuyer

Springer

Advances in Modeling and Simulation

Zdravko Botev · Alexander Keller ·
Christiane Lemieux · Bruno Tuffin
Editors

Advances in Modeling and Simulation

Festschrift for Pierre L'Ecuyer

Springer

Editors
Zdravko Botev
Randwick, NSW, Australia

Christiane Lemieux
University of Waterloo
Waterloo, ON, Canada

Alexander Keller
NVIDIA
Berlin, Germany

Bruno Tuffin
Rennes Bretagne-Atlantique
INRIA
Rennes Cedex, France

ISBN 978-3-031-10192-2 ISBN 978-3-031-10193-9 (eBook)
https://doi.org/10.1007/978-3-031-10193-9

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

This Festschrift is a collection of invited research articles on the occasion of Pierre L'Ecuyer's 70th birthday in 2020. During the pandemic, a celebration with friends and colleagues was impossible. When discussing the opportunity to publish a Festschrift instead, support has been enthusiastic. We are grateful to the authors of this volume for their endorsement and their ready willingness to contribute. The works reflect Pierre's influence on the fields of stochastic modeling, simulation, and operations research. It is a real pleasure to present this Festschrift to honor Pierre L'Ecuyer.

Sydney, Australia — Zdravko Botev
Berlin, Germany — Alexander Keller
Waterloo, Canada — Christiane Lemieux
Rennes, France — Bruno Tuffin
April 2022

Acknowledgements

We express our gratitude to Martin Peters at Springer-Verlag for supporting and publishing the Festschrift volume.

The manuscripts in this volume were carefully screened. We thank the anonymous reviewers for their time and their reports that contributed enormously to the excellent quality of the Festschrift.

Zdravko Botev
Alexander Keller
Christiane Lemieux
Bruno Tuffin

Biography

Abstract Pierre L'Ecuyer is regarded as a top scientist, his leadership in the field of simulation and, in particular, pseudo-random number generation being uncontested. Pierre is not only known for his scientific results, but also for his rigorousness, his dedication to excellence, his dynamism, his enormous working capacity, and his curiosity making an impression on every person who gets to know him. Those in desire of more detail may have a look at Pierre's vita at http://www.iro.umontreal.ca/lecuyer/cva.pdf.

Education

Pierre followed an education path from the Université de Montréal, Canada. His first academic degree is a Bachelor's in mathematics in 1972, followed by an M.Sc. in Operations Research in 1980 (in between he had been a teacher of mathematics at the CEGEP of Sept-Iles, Québec), and then a Ph.D. in Computer Science with Operations Research orientation in 1983 on Markovian decision processes. From 1982 to 1990, he served as a Professor in the Computer Science Department at Université Laval, and since then has been a Professor in the "Département d'informatique et de recherche opérationnelle" (DIRO) at the Université de Montréal. Pierre has also been spending time visiting colleagues in numerous places worldwide during sabbaticals or long stays, including Stanford University, Université de Nantes, Waseda University, University of Salzburg, North Carolina State University, Université de Savoie, Inria Rocquencourt and Rennes, University of New South Wales, and Google Research.

Research Activity and Visibility

As of the beginning of November 2021, Pierre has written or co-written 26 book chapters, 125 journal papers (in prestigious journals), and 134 referred conference papers. According to Google Scholar, his H-index is 68 and he has 16,481 citations. Pierre has a wide range of interests and diverse contributions. Among these, Pierre is a world leader in multiple areas:

- **Dynamic programming and operations research (OR) in general**. Pierre's activity on dynamic programming (starting with his Ph.D. thesis) and a broad range of OR domains, including the computation of derivatives, is still highly referenced in the domain.
- **Variance reduction techniques**. His work on variance reduction, including rare event simulation, has made Pierre one of the most renowned researchers in the domain, both from the theoretical and application (to reliability and queuing) aspects. He contributed to the advances of importance sampling and splitting procedures drastically reducing the simulation time to reach a predefined accuracy.
- **Telephone call centers**. His industrial contracts have led Pierre to work on the modeling, simulation, and optimization of call centers. His activity has led to the development of a software used by several companies and the development of models and specific analysis techniques making him a renowned expert in the area collaborating with the best-known other teams in the world. He notably developed new and more realistic models than those existing at the time in which the arrival rate of customers changes with time, is stochastic, and the arrival rates in different time periods are not independent. Novel estimations of parameters have also been designed. He also developed simulation-based optimization algorithms and heuristics for agent's staffing.
- **Quasi-Monte Carlo methods** are deterministic methods, as opposed to the random Monte Carlo ones, having the advantage of converging faster, even if less easy to apply. Again, Pierre has developed a strong and world-leading expertise in the generation of sequences of highly uniformly distributed points used by those methods, their randomization to get a practical estimation of the error, and their application in finance.
- **Random number generators**. While Pierre's work on all the previously described topics is already impressive, his activity on random number generators, a key issue for simulation, is probably what should be highlighted the most. If one has to give a single name on this topic, Pierre is probably the one that will be mentioned. His random generator RNGStreams is widely used, because it is one of the most efficient and portable ones. His extensive test suites are also very popular. Pierre's TestU01 software library is the standard suite of procedures for empirically testing the performance of random number generators. Among his many publications on the subject, the one he has in the *Communications of the ACM* has been cited more than 1,100 times.

Editorial Activities

Pierre is currently an Associate Editor for three journals: *ACM Transactions on Mathematical Software, Statistics and Computing*, and *International Transactions in Operational Research.* He was previously Associate Editor for five other journals and the Departmental Editor for the Simulation Department of Management Science. He was the Editor-in-Chief for *ACM Transactions on Modeling and Computer Simulation* from 2010 to 2013, a period during which the journal grew in scope and volume of submissions; Pierre was carefully reading all the submitted papers and prescreening submissions to alleviate the workload of the editorial team.

Remarkably, Pierre has reviewed articles for 170 *different* journals. That illustrates how well-known he is by people even outside the simulation community. Typically, he has been reviewing between three to four papers *per week* on average.

Organizing Conferences

Pierre has already organized seven international events in Montréal, including the Eighth International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing (MCQMC) in 2008 and the INFORMS Simulation Society Workshop in 2011, or the Eleventh International Conference on Monte Carlo Methods and Application (MCM) in 2017. He also co-organized MCQMC 2018 in Rennes, France. Pierre has been serving on many program or steering committees.

Simulation Societies

Pierre has been a member of numerous evaluation committees worldwide for grant proposals, promotion, and prizes from universities. Among the most notable memberships are the INFORMS Simulation Society Distinguished Service Award, INFORMS Simulation Society Outstanding Publication Award, or INFORMS College on Simulation Outstanding Publication and Outstanding Award committees.

Industry and Civil Society

Pierre has received many grants from the industry to successfully apply his simulation and operations research results. For example, Pierre has developed specific modeling and simulation tools for Bell and Hydro-Québec. He has also been contacted to implement his random number generators, or test existing ones, by AMD, Alcatel, LottoQuébec, The Mathworks, and Montréal Police Service, to name a few.

Pierre has devoted a lot of time to software development, with all his random generators available in many languages, a Java library for stochastic simulation called SSJ, or a software library called TestU01 offering a collection of utilities for the empirical statistical testing of uniform random number generators. All these tools are freely available to the scientific community.

Mentorship

Pierre has supervised 49 Master and Ph.D. students, as well as 23 postdocs. He has been a member of numerous Ph.D. examination committees all over the world. His courses on simulation at the Université de Montréal and various Summer Schools have also contributed to the widespread dissemination of modeling and simulation knowledge.

Recognition

Pierre has been recognized by the scientific community, having received prestigious awards such as a Canadian and an Inria Research Chair, the Award of Merit from the Canadian Operational Research Society, the INFORMS Fellow Award in 2006, the INFORMS Simulation Society Distinguished Service Award in 2011, the INFORMS Simulation Society Outstanding Research Publication Award won three times in 1999 (on Combined Multiple Recursive Random Number Generators), 2009 (on computational finance, by designing efficient algorithms for pricing path-dependent options), and 2018 (on call centers modeling), the SIGSIM Distinguished Contributions Award in 2016, or INFORMS Simulation Society Lifetime Professional Achievement Award in 2020. On the Canadian scene, to name only a few of the awards and distinctions earned by Pierre, in 1996, he was awarded a prestigious Steacie Fellowship for the period 1995–1997 from the Natural Science and Engineering Research Council of Canada. He received a Killam Research Fellowship from the Canada Council for the Arts for the period 2001–2003. And in 2004, he was awarded a Canada Research Chair on Stochastic Simulation and Optimization for the period 2004–2010, which was renewed in 2011 for another seven years.

Personal Achievements

Besides his scientific life, Pierre has had an amazing sport-related life, for which he is also well-known in the respective communities: not only as a competitor but also as a coach. While skilled in many kinds of sport, Pierre is exceptionally competitive and skilled in cross-country skiing and road cycling.

In cross-country skiing, he won the bronze medal at the Canadian championship in 50 km classic in 1994. But his main achievements are probably in road cycling. He was Canadian champion (by age groups) in 2000, 2001, 2011, and 2012 and finished second in 2002 and 2004. He was Quebec champion in road race in 1996, 2000, 2001, 2003, and 2012; in time trial in 2002, 2003, 2004, 2005, and 2012; and in criterium in 2011 and 2016. Among successful participation in other races, he won the America's cup in 2000 and 2012. He was named cyclist of the year 2012 in masters categories by the Quebec Cycling Federation, another award in another category! Pierre keeps on riding his bicycle all over the world and is known to always bring his bike with him when traveling.

Prior to these accomplishments, Pierre had been a coach in Track and Field between 1970 and 1992. He started by building and training a local team. He became a member of the Canadian team at the Olympic Games, World Championships, Commonwealth Games, etc. He was named Quebec's track and field federation's "coach of the year" in 1985 and 1992. He was the coach of the Barcelona Olympic games silver medalist (20 km walk) and world record holder (30 km walk) Guillaume Leblanc, from 1973 to 1992.

Contents

Monte Carlo Methods for Pricing American Options

Raul Chavez Aquino, Fabian Bastin, Maria Benazzouz, and Mohamed Kharrat

Abstract American options are widespread in the financial market. We review various popular techniques used to value American options, as well as Malliavin calculus and recent approaches proposed in machine learning, and examine their performance on synthetic and real data. Our preliminary results confirm that pricing an American put option on a single asset can be efficiently done using regression approaches, and random forests are competitive in terms of accuracy and computation times. Malliavin calculus, despite its interesting mathematical properties, is not competitive for American option pricing, and neural networks are difficult to design in the context of options. Variance reduction, achieved here by means of control variates, is a crucial tool to obtain reliable results at a reasonable cost.

Keywords American options · Monte Carlo · Dynamic programming · Variance reduction · Control variates

R. Chavez Aquino · M. Benazzouz
Department of Economics, Université de Montréal, Montréal, QC, Canada
e-mail: raul.chavez.aquino@umontreal.ca

F. Bastin (✉)
Department of Computer Science and Operations Research, Université de Montréal, and CIRRELT, Montréal, QC, Canada
e-mail: bastin@iro.umontreal.ca

M. Benazzouz
Desjardins, Laval, QC, Canada

M. Kharrat
Department of Mathematics, Jouf University, Sakaka, Saudi Arabia
e-mail: mohamed.kharrat@fphm.rnu.tn

Laboratory of Probability and Statistics LR18ES28, Sfax University, Sfax, Tunisia

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_1

1 Introduction

Since their official recognition as a financial tool in 1973 with the creation of the Chicago Board Options Exchange, options have attracted a lot of interest by investors, traders and academicians. An option gives the right, but not the obligation, to buy (call option) or sell (put option) an asset at a predetermined price, either at a fixed time T, called the maturity, or at times in a set $\mathcal{T}$, but no later than T. An American option is defined as an option giving its holder the privilege to exercise it at any time during its life. In order to benefit from this privilege, the holder of this type of option must exercise at the best possible time. The option is usually based on underlying price series (stock price, interest rate, index value, etc.) whose random fluctuations are modeled using stochastic processes. The main difficulty with pricing American options is getting a reliable estimate of the continuation value.

One of the first techniques proposed to valuate financial options is the binomial tree, introduced by Cox et al. [16] and Rendleman and Bartter [40]. Binomial trees rely on the discretization of the stochastic differential equation governing the assets value evolution, and do not generalize well to high dimensional portfolios and stochastic processes other than the geometric Brownian motion. An alternative, introduced by Boyle [10] in the context of option pricing, is the Monte Carlo simulation of asset price trajectories. The value of an American option at some time, given a scenario, is expressed as the maximum between the instantaneous exercise price and the expected continuation value, that is the optimal return that can be obtained if one delays the exercise at a later stage. In their seminal paper, Longstaff and Schwartz [33] use dynamic programming and Monte Carlo simulation to estimate the optimal stopping time frontier, relying on least squares regression to approximate the continuation value. Clément et al. [15] analyze the theoretical properties of the method, that can provide lower bounds in expectation, close to the true option prices.

The continuation value can also be estimated with other techniques. Capitalizing on the formulation proposed by Lions and Régnier [32], Bally et al. [3] explore the use of Malliavin calculus [34, 35]. Their work is further explored by Caramellino and Zanette [13], stressing on the importance of an indirect variable, while Kharrat and Bastin [28] develop the expression of Malliavin weights for American options under a stochastic volatility. Bouchard and Warin [9] compare regression approaches and Malliavin calculus, but without variance reduction techniques. The use of Malliavin calculus in financial engineering had recently been covered by Alòs and Lorite [1], but without consideration of options, and we refer to Pascucci [37] for a more specific coverage. Ruf and Wang [41] review the use of neural networks for option pricing, while Rabia [38] considers the use of random forests, reporting promising results.

All these approaches can greatly benefit from the use of variance reduction techniques, although many of the previously cited works do not take full advantage of them. In particular, Rasmussen [39] proposes to use European options, valued at the exercise times, as control variates, and reports significant improvement in the option valuation accuracy when combined with the Longstaff and Schwartz's approach. Dion and L'Ecuyer [17] explore the use of quasi-Monte Carlo draws. Another strat-

egy to reduce the variance is the multilevel Monte Carlo technique, proposed by Giles [20]. The main idea is to sum estimators built on multiple sets of simulations with different time steps. Such estimators can be obtained using any of the reviewed dynamic programming approaches. Mixed results have however been reported for American options (see for instance Wu [43]), and Belomestny et al. [5] suggest to rather consider levels corresponding to different degrees of approximation of the continuation values. They illustrate the method on a multi-assets option, using the mesh method [12] and the regression approach [33], in combination with the simpler control variate proposed by [12], reporting promising results along with complexity reduction guarantees.

In this chapter, we numerically compare these techniques to valuate an American option in a simple, yet reasonable, assumptions framework, and derive some practical guidelines, emphasizing the importance of variance reduction techniques. Other techniques have been proposed to price American options, in particular the stochastic mesh method [12]. However, regression methods as the one proposed by Longstaff and Schwartz, when combined to variance reduction techniques, have been shown to outperform the stochastic mesh method, in accuracy and computational time [42]. We will therefore not consider such approaches.

The rest of this chapter is organized as follows. We introduce the American option pricing problem in Sect. 2, and present the binomial tree method in Sect. 3. Section 4 covers dynamic programming approaches and we discuss the use of control variates in Sect. 5. Numerical comparisons are performed in Sect. 6. We conclude and present future research avenues in Sect. 7.

2 American Option Pricing

The problem of pricing, or valuing, an American option consists of finding an optimal exercise strategy and valuing the expected discounted payoff from this strategy. Given the underlying asset value S_t, let denote by V_t the option value at time t, defined as

$$\beta_t V_t(S_t) = \sup_{\tau \in \mathcal{T}(t,T)} \mathbb{E}_t[\beta_\tau X_\tau(S_\tau) \mid S_t], \tag{1}$$

where $\{X_t\}_{0 \le t \le T}$ is the payoff process with the discount factors β_t, $t \in [0, T]$, with $\beta_0 = 1$, and $\mathcal{T}(t, T)$ denotes the class of stopping times satisfying $t \le \tau \le T$. The American option value is defined as V_0, i.e. the value at time 0.

We assume that the factors β_t, $t \in [0, T]$ are deterministic and the asset value process $\{S_t\}_{0 \le t \le T}$ is Markovian and exogenous, i.e. is not affected by the decision to exercise the option or to wait. For simplicity, we only consider an option on a single asset without dividends. The exercise price at time t of a put option is

$$X_t(S_t) = (K - S_t)^+ := \max\{K - S_t, 0\},$$

where K is the strike. For a call option, the exercise price is $\max(S_t - K)^+$. If the discount factors β_t, $t > 0$, are less than one, it can be shown that the optimal exercise of an American call option is at the expiration date T [36]. The option is then equivalent to a European call option with maturity date T, whose value at time t is

$$\beta_t V_t^E(S_t) = \beta_T \mathbb{E}[X_T(S_T) \mid S_t],$$

and we will focus on put options. We can similarly define a European put option by allowing to exercise the option at time T only. Obviously, $V_0^E(S_0) \leq V_0(S_0)$.

Solving (1) is usually intractable, and the problem is simplified by restricting the set of possible exercise times. A usual approximation is obtained with the Bermudan option, allowing the option holder to exercise it at equidistant times $t_m = m\Delta t$, for $m = 0, \ldots, M$, and $\Delta t = T/M$. Denoting $\mathcal{T}_B(0, T)$ the class of stopping times satisfying $\tau = m\Delta t, m \in \{0, \ldots, M\}$, the value of the Bermudan option is

$$V_0^B(S_0) = \sup_{\tau \in \mathcal{T}_B(0,T)} \mathbb{E}[\beta_\tau X_\tau(S_\tau) \mid S_0] \leq V_0(S_0).$$

Under mild conditions, $V_0^B(S_0)$ converges to $V_0(S_0)$ when m grows to infinity.

It is also often important to measure the sensitivity of the option price with respect to some initial conditions. The most common is the Delta, that measures the change in the option price as a result of a change in the value S_0 of the underlying. The other options valuation variables being constant:

$$\Delta(S_0) = \frac{\partial}{\partial S_0} V_0(S_0),$$

where we have stressed the dependency of the option value towards the initial asset value. All the techniques covered in this chapter can be adapted to compute the Delta, that can be approximated by finite difference, using common random numbers for the simulation-based methods covered in Sect. 4. While we have computed them during our numerical experiments, we do not report their value for conciseness.

In the simplest case, the asset value S_t is often assumed to follow the Black-Scholes model, that can be described with the partial differential equation

$$dS_t = rS_t dt + \sigma S_t dB_t, \tag{2}$$

where r is the risk-free interest rate, σ is the volatility, and B_t is a standard Brownian motion, and a known initial asset value $S_0 > 0$. The asset price at time t can also be expressed as

$$S_t = S_0 \exp(ht + \sigma B_t)$$

where $h = r - \frac{1}{2}\sigma^2$. Under the Black-Scholes model, it is possible to derive analytical expressions of the price of simple options, as the European option (see for instance Higham [23, Chap. 8]). Moreover, we have

$$\beta_t = e^{-rt}, \quad \beta_{\Delta t_i} := \beta_{\Delta t} = e^{-r\Delta t}, \text{ for } i = 0, \ldots, M-1.$$

Given the time discretization $\{t_i = i\Delta t,\ i = 0, \ldots, M\}$, we define the continuation value at time t_i as the discounted conditional expectation $\beta_{\Delta t_i}\mathbb{E}\left[V_{i+1}(S_{i+1}) \mid S_t\right]$, where $V_i(S_i)$ is defined as in (1). As the asset value process is Markovian and exogenous, if the option has not yet been exercised, the option value at time t_i is

$$V_{t_i}(S_{t_i}) = \max\left\{X_{t_i}(S_{t_i}), \beta_{\Delta t_i}\mathbb{E}\left[V_{t_{i+1}}(S_{t_{i+1}}) \mid S_{t_i}\right]\right\}. \tag{3}$$

The rationale is to exercise if the exercise price surpasses the continuation value. However, the continuation value has no analytical expression and must be approximated. We now review several approaches and compare them on simple examples.

3 Binomial Tree Method

A simple approach to price a single-asset option is the binomial model, introduced by Cox et al. [16]. At each time step, we assume that the asset value can either go up by a factor u, with a probability p, or down by a factor d, with a probability $1-p$. This allows to represent M scenarios with a recombined tree, as in Fig. 1. If the asset value follows the Black-Scholes model (2), the discretization of the process on the binomial tree with $p = \frac{1}{2}$ leads to set

$$u = exp\left(\sigma\sqrt{\Delta t} + \left(r - \frac{\sigma^2}{2}\right)\Delta t\right), \quad d = exp\left(-\sigma\sqrt{\Delta t} + \left(r - \frac{\sigma^2}{2}\right)\Delta t\right)$$

(see for instance Higham [23, Chap. 16]). It is then easy to compute the value of the option on the discretized asset value process by proceeding backward on the binomial tree. At the expiration date, for $n = 1, \ldots, M$, we set $V_M^n = (K - S_M^n)^+$,

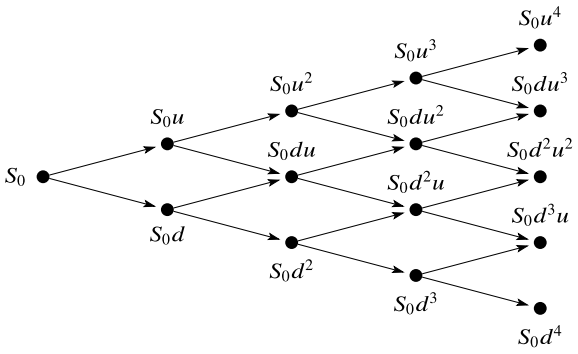


Fig. 1 Binomial tree

and then, we recursively compute, for $0 \leq n \leq i,\ i = 1, \ldots, M-1$,

$$V_n^i = \max\left\{\left(K - S_n^i\right)^+, \beta_{\Delta t}\left(pV_{n+1}^{i+1} + (1-p)V_n^{i+1}\right)\right\}. \tag{4}$$

Equation (4) can be seen as an approximation of (3), the continuation value being approximated as the discounted expected value of the two children nodes values. The American option value $V_0(S_0)$ is then approximated by V_0^0.

Several tricks have been proposed to improve the method accuracy. Here we follow the suggestions made by Broadie and Detemple [11] to construct the binomial Black-Scholes with Richardson extrapolation (BBSR) method. At the stage $M-1$, the Black-Scholes formula replaces the usual continuation value in (4) by

$$V_{M-1}^i = \max\left\{(K - S_{M-1}^i)^+, K\beta_{\Delta t}\Phi(-d_2) - S_{M-1}^i\Phi(-d_1)\right\}.$$

where $\Phi(\cdot)$ is the standard normal cumulative distribution function, with

$$d_1 = \frac{\ln\left(S_{M-1}^i/K\right) + \Delta t(r + \sigma^2/2)}{\sigma\sqrt{\Delta t}}, \quad d_2 = d_1 - \sigma\sqrt{\Delta t}.$$

They furthermore add a two-point Richardson extrapolation consisting in pricing the option with M and $2M$ time steps. Denote by V_M and V_{2M} the respective prices. The approximate option price is then set to $\hat{V}_0 = 2V_{2M} - V_M$.

Efforts have also been made to improve the accuracy by using more complex structures, in particular trinomial trees, but they have not delivered any advantage over binomial trees [14], and therefore are not considered here.

4 Dynamic Programming Approach

The binomial tree method provides a simple and effective method for put options on a single asset following the Black-Scholes model (2), and we will use it as a benchmark in our numerical experiments. However, the approach cannot be easily extended to put an option on multiple assets that are not subject to the Black-Scholes model. The other methods that we consider rely on the dynamic programming principle and Monte Carlo simulation [21]. The basic idea consists to sample N price trajectories from the initial asset value S_0, by drawing the asset value S_i^n at time t_i, producing the asset values $\{S_0, S_1^n, \ldots, S_{i-1}^n\}$, for $i = 1, \ldots, M$, $n = 1, \ldots, N$, and estimate the expected continuation values at each time step for each scenario. Given a scenario and a time stage i, we exercise the option if we obtain a better return value.

We first compute the option value at the expiration date T, for each trajectory $n = 1, \ldots, N$, as

$$\hat{V}_T^n(S_M^n) = \left(K - S_M^n\right)^+.$$

We then compute estimators of the option value at each time step for each simulated trajectory, using a backwards recursion. For $i = M-1, \ldots, 1$, $n = 1, \ldots, N$, we set

$$\hat{V}_{t_i}^n(S_i) = \begin{cases} X_{t_i}(S_i^n) & \text{if } X_{t_i}(S_i^n) \geq \tilde{V}_{t_i}(S_i^n), \\ \beta_{\Delta t_i} \hat{V}_{t_{i+1}}^n(S_i^n) & \text{otherwise,} \end{cases} \tag{5}$$

where $\tilde{V}_{t_i}(S_i)$ is an estimation of the continuation value at time t_i, given the current asset value S_i, and

$$\beta_{\Delta t_i} = \frac{\beta_{t_i}}{\beta_{t_{i+1}}}.$$

We finally build the estimator of the option value $V_0(S_0)$ as the maximum between the exercise price at time 0 and the empirical average of the estimated continuation values over the N simulated trajectories, as

$$\hat{V}(S_0) = \max \left\{ X_0(S_0), \beta_{t_1} \frac{1}{N} \sum_{n=1}^{N} \hat{V}_1^n \right\}. \tag{6}$$

The methods differ in the way we compute $\tilde{V}_{t_i}(S_i)$, $i = 1, \ldots, M-1$ in (5).

4.1 Regression Methods

Longstaff and Schwartz [33] propose to build a function predicting the continuation value by assuming that the latter can be expressed as a linear combination of a countable set of basis functions: for $i = 1, \ldots, M-1$,

$$\beta_{\Delta t} \mathbb{E}\left[V_{t+\Delta t}(S_{i+1}) \mid S_i \right] = \sum_{j=1}^{\infty} \alpha_t^j F_t^j(S_i). \tag{7}$$

In order to make it computationally feasible, the sum in (7) is truncated to the J first terms, assuming that the truncation error can be neglected:

$$\beta_{\Delta t} \mathbb{E}\left[V_{t+\Delta t}(S_{i+1}) \mid S_i \right] \approx \sum_{j=1}^{J} \alpha_t^j F_t^j(S_i). \tag{8}$$

The weights α_t^j are then estimated using a linear regression, i.e. by minimizing the mean square error

$$\min_{\alpha_t} \frac{1}{\#\mathcal{N}_t} \sum_{n \in \mathcal{N}} \left(\sum_{j=1}^{J} \alpha_t^j F_t^j \left(S_t^n\right) - \beta_{\Delta t} \hat{V}_{t+\Delta t}^n \left(S_{t+\Delta t}^n\right) \right)^2, \tag{9}$$

where $\mathcal{N}_t$ is the subset of the N simulated trajectories for which the position at time t is in the money, i.e. for any $n \in \mathcal{N}_t$, $K > S_t^n$, while for any $n \in N \setminus \mathcal{N}_t$, $K \leq S_t^n$. This limits the size or the problem to solve, but also reflects that for trajectories out or at the money at time t, it is always preferable to wait rather than to exercise the option. Various choices can be made for the basis functions [39]. The simplest is to fit a multivariate polynomial in the case of an option over a multidimensional portfolio [21]. For an option over a single asset, we will simply fit a polynomial of degree d.

The continuation value can also be estimated by means of machine learning regression techniques. We first consider the random forest method, a supervised machine learning algorithm widely used in classification and regression problems and initially introduced by Ho [24]. We here focus on the main ingredients, referring to Hastie et al. [22, Chap. 15] for more details. At each time step t_i, $i = 1, \ldots, M-1$, we generate K bootstrap samples, and for each of them, we construct a regression tree [27, Chap. 8], aiming to predict the value of the dependent variable, here the continuation value, as a function of the independent variable, the asset value for each in-the-money trajectory at the time $t_i + \Delta t$. The approach is similar to the expression (8), but with a non-parametric model, the weights being replaced by the tree structure. First, a large tree is obtained by means of recursive binary splitting, as each non-terminal node $\mathfrak{n}$ is associated with to a subset of the asset values at time t_i, and is split into two children nodes, gathering the values either less or greater than some value $s_{\mathfrak{n}}$, typically the mean of the values associated with to the father node. We stop when each terminal node has less than some minimum number of observations. The tree size is then reduced by applying cost complexity pruning, in order to limit the number of leaf nodes, knowing that more nodes tend to provide a better estimation, but a greater risk of overfitting. Each terminal node then provides a prediction of the dependent variable value, i.e. the continuation value, and the predictions are averaged to produce the output of the tree. We can then use this built collection of trees to compute a set of predictions $\mathfrak{v}_{t_i}^k(S)$, $k = 1, \ldots, K$, given the asset value S, and average them to produce the estimation of the continuation value as

$$\tilde{V}_{t_i}(S) = \frac{1}{K} \sum_{k=1}^{K} \mathfrak{v}_{t_i}^k(S).$$

Another class of models popular in machine learning are neural networks. A neural network is a directed graph where each node represents a neuron, that receives a number of inputs x_i from its direct predecessors and applies an activation function to produce an output that feeds its direct successors. Several activation functions exist. We here choose the ReLU function, defined as $ReLU(x) = x^+$, as it guarantees that only non-negative values are produced. We also limit ourselves to multilayer

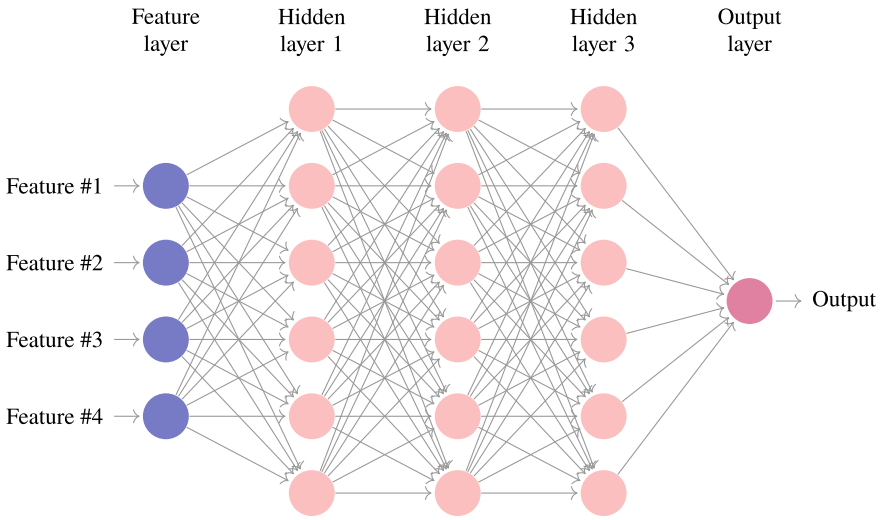


Fig. 2 The architecture of a neural network

perceptions (MLP) [22, Chap. 11], where each neuron receives the output of the previous layer as input, and its own output is the input of the next layer, as illustrated in Fig. 2.

At each stage t_i, $i = 1, \ldots, M-1$, we train a neural network by minimizing a loss function, here defined similarly to (9) as the mean square error minimization problem

$$\min_{\gamma_{t_i}} \frac{1}{\#\mathcal{N}_{t_i}} \sum_{n \in \mathcal{N}_{t_i}} \left(\tilde{V}_{t_i}\left(\gamma_{t_i}, S^n_{t_i}\right) - \beta_{\Delta t_i} \hat{V}^n_{t_i+\Delta t_i}\left(S^n_{t_i+\Delta t_i}\right) \right)^2, \tag{10}$$

for the $\#\mathcal{N}_{t_i}$ in-the-money scenarios at stage t_i. We then set

$$\tilde{V}_{t_i}(\cdot) = \tilde{V}_{t_i}(\gamma^*_{t_i}, \cdot),$$

where $\gamma^*_{t_i}$ is the optimal solution found when solving (10). The machine learning architectures have to be trained on a large number of scenarios (we consider 100000 scenarios in our experiments), leading to a significant overhead time. Once trained, they can be used on several replications of the option pricing problem, as long as the underlying asset value follows the same process.

4.2 Malliavin Calculus

Fournié et al. [19] express the expected value of the option at time t, conditionally to the asset value at time $s \leq t$, as

$$\mathbb{E}\left[V_t(S_t) \mid S_s = \alpha\right] = \frac{\mathbb{E}\left[\pi_{s,t}(V_t(S_t), \alpha)\right]}{\mathbb{E}[\pi_{s,t}(1, \alpha)]} \tag{11}$$

where $\pi_{s,t}(\cdot)$ are the Malliavin weights, whose expressions vary according to the asset value process. Under the Black-Scholes model (2), Bally et al. [2] show that

$$\pi_{s,t}(\xi, \alpha) = \frac{\xi H(S_s - \alpha)}{\sigma s(t-s) S_s} \Delta B_{s,t},$$

where $\Delta B_{s,t} = tB_s - sB_t + \sigma s(t-s)$, and $H(\cdot)$ is the Heaviside step function, defined as $H(\ell) = \mathbb{1}(\ell \geq 0)$, $\mathbb{1}$ representing the indicator function. They further refine the computation of the conditional expectation (11) by introducing a density function ψ parameterized by $x \geq 0$, referred as the localization function, leading to

$$\mathbb{E}[V_t(S_t) \mid S_s = \alpha] = \frac{\mathbb{T}^{\psi}_{s,t}(V_t(S_t), \alpha)}{\mathbb{T}^{\psi}_{s,t}(1, \alpha)} \tag{12}$$

with

$$\mathbb{T}^{\psi}_{s,t}(\xi, \alpha) = \mathbb{E}[\xi \psi(S_s - \alpha)] + \mathbb{E}\left[\xi \frac{H(S_s - \alpha) - \Psi(\xi, S_s - \alpha)}{\sigma s(t-s) S_s} \Delta B_{s,t}\right],$$

where $\Psi(x, \ell) = \int_{-\infty}^{\ell} \psi(x, y) dy$ is the cumulative distribution function associated to $\psi(x, \cdot)$. We follow their suggestion and consider in our numerical experimentation the Laplace-type probability distribution function

$$\psi(x, y) = \frac{\lambda(x)}{2} e^{-\lambda(x)\ |y|} \tag{13}$$

where $\lambda(x) = 1/\sqrt{t-s}$.

Bally et al. [3] adapt the algorithm proposed by Lions and Régnier [32] and using a backwards recursion, they approximate the N continuation values $\mathbb{E}[V_{t_{i+1}}(S_{i+1}) \mid S_i = S_i^n]$, $n = 1, \ldots, N$, at time t_i, by taking the empirical sample averages of the expectations appearing in the right hand sides of (11) and (12) over the N simulated asset price trajectories. Ignoring the localization function for simplicity of exposition, the estimated discounted continuation value is

$$\tilde{V}_{t_i}(S_i^n) = \frac{\hat{\pi}_{t_i, t_{i+1}}(\hat{V}_{t_{i+1}}, S_i^n)}{\hat{\pi}_{t_i, t_{i+1}}(1, S_i^n)},$$

with

$$\hat{\pi}_{t_i,t_{i+1}}(\hat{V}_{t_{i+1}}, S_i^n) = \frac{1}{N}\sum_{q=1}^{N} \frac{\hat{V}_{t_{i+1}}^q(S_{i+1}^q) H\left(S_i^q - S_i^n\right)}{\sigma t_i \Delta t S_i^q} \Delta B_{t_i,t_{i+1}}^q,$$

$$\hat{\pi}_{t_i,t_{i+1}}(1, S_i^n) = \frac{1}{N}\sum_{q=1}^{N} \frac{H\left(S_i^q - S_i^n\right)}{\sigma t_i \Delta t S_i^q} \Delta B_{t_i,t_{i+1}}^q,$$

where $\Delta B_{t_i,t_{i+1}}^q$ is obtained from the draws generating the q^{th} asset value path. The computational effort required at each time step is therefore of order $O(N^2)$.

5 Control Variates

The convergence properties of the Longstaff and Schwartz's algorithm have been studied in details [15]. In essence, when N and M grow to infinity, (6) converges to a lower bound of the American option value. The quality of this bound can be refined and made arbitrarily close to the true value by improving the accuracy of the approximations $\tilde{V}_t(\cdot), t \in cT$, especially by letting J rise to infinity in (8). However, for finite N, M, and J, the error can be large and the estimator (6) can present a significant noise. If M is too small, the number of opportunities to exercise the option is limited and we tend to underestimate the option value, while if N is not large enough, we face the risk of overfit the policy with respect to the simulated trajectories, and to overestimate the option value. The quality of the estimator (6) can be improved by applying variance reduction techniques [6, 30], among which control variates have received a lot of attention for option pricing. Given an estimator X, the basic idea is to find an random variable Y whose expectation is known, and highly correlated with X. We then form the new estimator $Z = X - \theta(Y - E[Y])$. If θ is fixed, $E[Z] = E[X]$ and $\text{Var}[Z] = \text{Var}[X] + \text{Var}[Y] - 2\theta\,\text{Cov}(X, Y)$. The variance of Z can be minimized by choosing

$$\theta = \frac{\text{Cov}(X, Y)}{\text{Var}[Y]}.$$

θ can be estimated using pilot experiments, but more often, an estimator will be produced using the same draws as those used to generate X and Y, so that Z is biased, but usually, this bias is negligible. When θ is set to one, we say that Z is an indirect estimator, noticing that several authors still speak about control variate [3, 13, 25].

It is often possible to accurately estimate the price of a European option. When the asset value follows a simple process, as in the Black-Scholes model, it is even possible to derive it analytically. This suggests to use the European price estimator $\hat{V}_0^E = \beta_T(K - S_T)^+$ as a control variate for the American option price estimator [25]. Its performance is however disappointing, and several authors have searched for better candidates [8, 18]. Rasmussen [39] notices that an early exercise produces a

value whose correlation with the corresponding European option price degrades as the expiration date is later in the future. He then suggested to use as a control variate for the continuation value a European option emitted at the candidate exercise time of the American option. Indeed, the optional sampling theorem establishes that for any scenario n, the exercise value at T is an unbiased estimator of the price of the European option emitted at the exercise time τ_n of the American option under this scenario, and at time 0 if the American option is never exercised.

In order to apply this control variate, we first initialize the estimator of the European option at the expiration date T as $V_T^{E,n} = X_T^n(S_T^n)$. Going backwards, for $i = M-1, \ldots, 1$, in addition to $\tilde{V}_{t_i}(S_i^n)$, we produce an estimator $\tilde{V}_{t_i}^E(S_i^n)$ of the value of the European option emitted at t_i and expiring at T, for $n = 1, \ldots, N$, using the same estimation technique. We also estimate the second moment of this estimator and its covariance with the continuation value estimator, in order to get $\theta_{t_i}^n$. We then replace $\tilde{V}_{t_i}(S_i^n)$ by $\tilde{V}_{t_i}(S_i^n) - \theta_{t_i}^n(\tilde{V}_{t_i}^E(S_i^n) - V_{t_i}^E(S_i^n))$ to estimate the continuation value at time t_i for the scenario n. At time 0, the European option theoretical price is computed for each scenario, using the exercise time of the American option as the emission date of the European option, and a time 0 if is never exercised, leading to the new American option price estimator

$$\hat{Z}_0 = \max\left\{X_0(S_0), \frac{1}{N}\sum_{n=1}^{N}\left(\beta_{t_1}\hat{V}_1^n - \theta_0\left(\beta_{\tau_n}V_{\tau_n}^{E,n}(S_{\tau_n}^n) - V_0^E(S_0)\right)\right)\right\},$$

where τ_n is set to T if the option is never exercised in the scenario n. For more implementation details, we refer the reader to West [42, Chap. 3].

6 Numerical Experiments

We now evaluate the presented methods to estimate the price of American options. Similar results have been obtained for the Delta, but are not reported as they bring similar conclusions. As a convention, we abbreviate the method name by LS for the Longstaff and Schwartz's algorithm, RF for the Random Forests, NN for the Neural Networks, and Malliavin for the estimation based on Malliavin calculus. The experiments have been conducted using Python 3.9 and numpy and sklearn libraries, on an Intel i7-9700K at 3.60GHz, eight cores, and 32Go of memory. Random numbers have produced using the xoshiro pseudo-random generator [7]. Similar results have been obtained with the MRG32k3a generator [29]. We followed Bally et al. [2] when implementing the Malliavin calculus approach, using the localization function (13) and a pointwise indirect estimation for the computation of the continuation value, based on the European option value emitted at the exercise time. Results obtained with indirect estimation are identified with the postfix "-I", and with the suffix "-CV" for those produced with the Rasmussen's control variates. The implementation

code is available at https://github.com/RaulChavezAquino/Monte-Carlo-methods-for-pricing-American-Options.

We first consider an American put option on the stock price process S with parameters $S_0 = 100$, $K = 100$, $\sigma = 0.20$, $r = \ln(1.1)$, $T = 1$, taken from Bally et al. [3]. The binomial tree gives us a reference option value of 4.9175.

We report the estimation time as a function of M and N for each technique in Fig. 3. The MLP models have been trained using sklearn and the solver ADAM. For each time discretization factor $M = (10, 20, 50, 100, 500)$, we use (2, 2, 2, 3, 3, 3) hidden layers and (10, 20, 50, 25, 50, 70) neurons per layer, respectively, those configurations having been selected by trial and error. The random forests have been trained with 10 trees and a maximum number of leaves equal to 15. The machine learning models have been trained by 100000 scenarios, but the training time is not taken into account in these graphs, while it affects the overall performance. For the random forests, we observed an overhead due to the training going from 10 s for $(M, N) = (10, 500)$ to 235 s for $(M, N) = (200, 10000)$. We have also empirically observed that this overhead approximately grows linearly with M, but does not significantly change with N. MLP models required a large training time, and we stopped the computations when it exceeded 2 h. As a result, we only report results for $M = 10$ and $M = 20$ with the neural networks. From Fig. 3, we observe that LS is the fastest method while Malliavin calculus requires prohibitive times. The effect could be exacerbated due to the use of Python, an interpreted programming language, but we observe that the computation time rises at a rate faster than a linear rate with the number of scenarios, as expected, while the other approaches exhibit an computation time that grows approximately linearly with N.

We next analyze in more details the performances of the methods when we vary N and M, for the naive estimator, the indirect estimator, and the estimator with the control variate. For each configuration, we repeat the valuation 1000 times, and draw a box plot over these replications. Figures 4 and 5 show the behavior of LS and RF methods, respectively, where we fix $N = 5000$ on the left part and $M = 100$ on the right part. When M is small and N is kept constant, we tend to underestimate the option value, but we can face some overfit when M is large with LS. On the other side, fixing M reveals a large overfit for small values of N with LS, but a convergence to a lower bound of the option price when N grows. Several configurations were tried with the random forests. The two upper graphs in Fig. 5 have been obtained with 50 trees and a maximum number of leaves of 20, while the two lower graphs have been computed with 90 trees and a maximum of 100 leaves. We see that the random forests always underestimate the true price, and the effect can be dramatic with the naive estimator when we have many leaves. However, applying variance reduction techniques allow to obtain good price estimators, and the estimators obtained with 100 leaves are close to the ones obtained with the least-squares approach. Large random forests therefore appear very sensitive to the variance in the asset price trajectories, and more research is needed to properly explain the method behavior, and to automatically choose the best design. Reducing the variance nevertheless allows the random forests to be competitive, and they represent an alternative method to parametric regression that we plan to further investigate. Of course, the option value

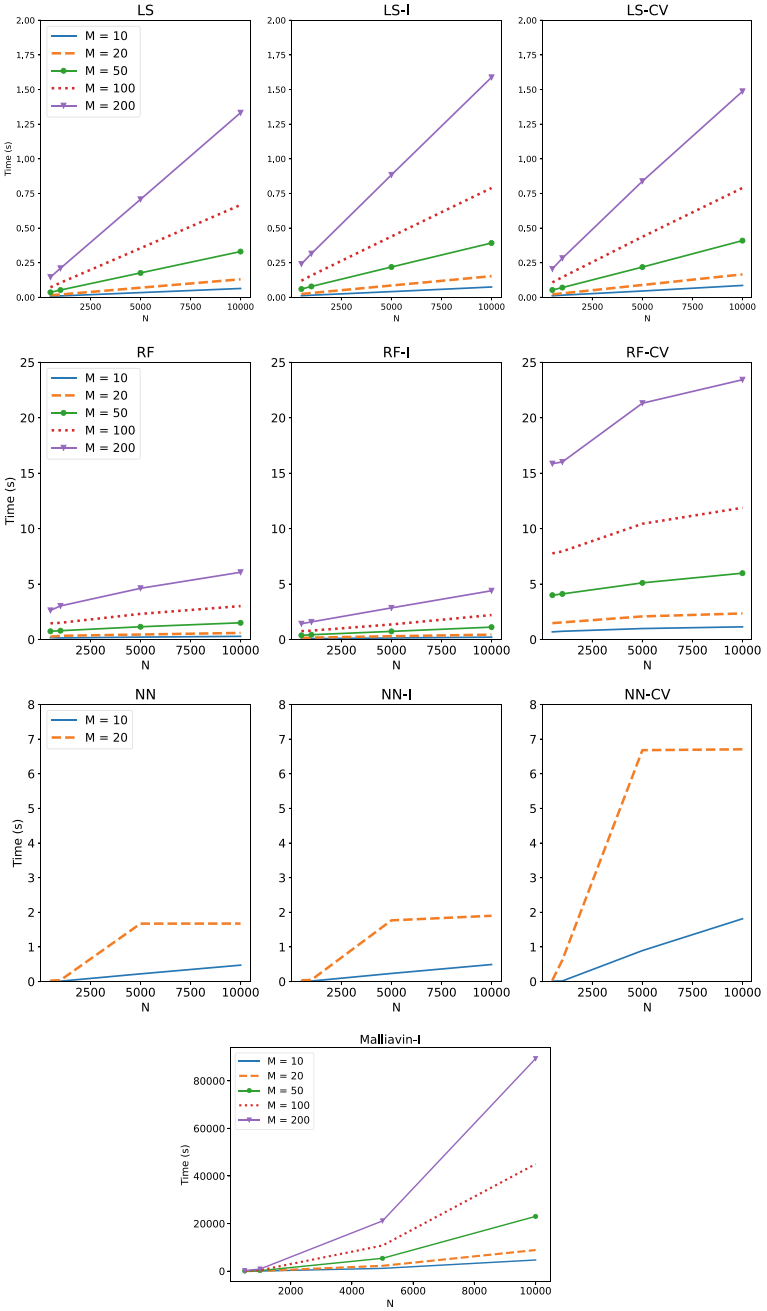


Fig. 3 Estimation time (*s*)

can also be computed during the training phase of the model, and for large M and N, random forests deliver good price estimation, even with a simple configuration of 10 trees and a maximum of 15 nodes used to produce the results reported in Table 1, computed over one run, where we additionally report in brackets the standard deviation over the N simulated trajectories.

We finally illustrate the MLP model with $N = 1000$, $M \in \{10, 20, 50, 100, 200\}$, and the Malliavin calculus, with $M = 100$, and N going from $N = 100$ to 20000, in Fig. 6. In both cases, the computational burden prevented us to perform more experiments. We first observe that the MLP model strongly underestimates the option value, the control variate performing better than the naive estimator, but less than the indirect estimation, that is still far from the quality obtained with the LS and RF approaches. Better MLP designs could improve the results, but we do not have an automatic way to find a good MLP structure. On the other hand, the Malliavin calculus gives good price estimation when the number of scenarios is at least equal

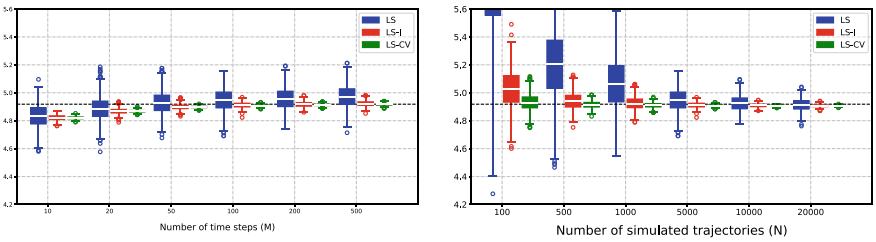


Fig. 4 Least-squares regression

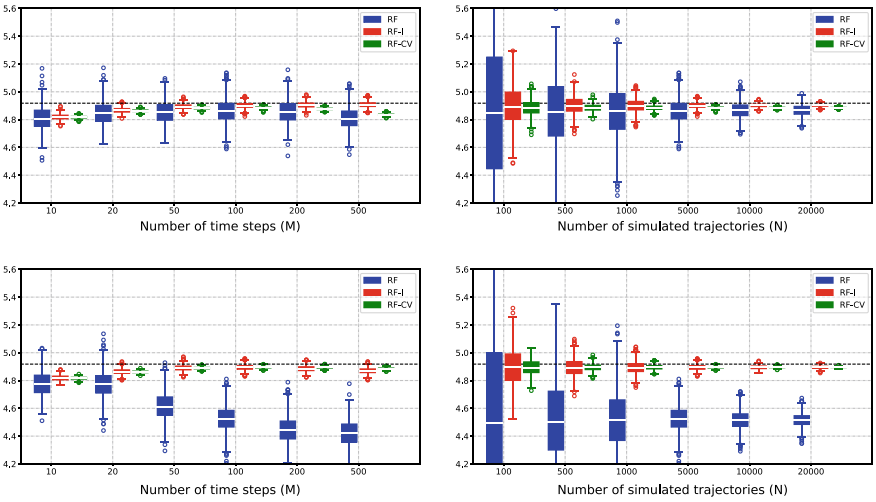


Fig. 5 Random forests

Table 1 American option value with RF

M	N	RF	RF-I	RF-CV
10	10000	4.89 (6.01)	4.82 (1.36)	4.82 (0.65)
	100000	4.71 (5.99)	4.82 (1.36)	4.82 (0.64)
20	10000	4.90 (5.99)	4.87 (1.38)	4.86 (0.60)
	100000	4.77 (5.91)	4.86 (1.38)	4.86 (0.61)
50	10000	4.98 (5.97)	4.92 (1.40)	4.89 (0.58)
	100000	4.85 (5.89)	4.91 (1.41)	4.89 (0.58)
200	10000	5.04 (5.81)	4.92 (1.38)	4.90 (0.56)
	100000	4.89 (5.68)	4.92 (1.37)	4.90 (0.55)

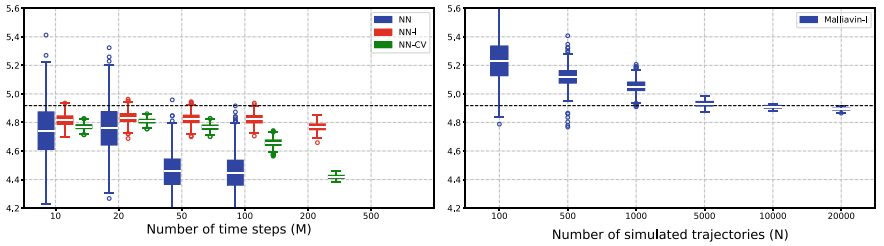


Fig. 6 Neural networks (left) and Malliavin-I (right)

to 5000. However, the LS and RF models already perform well with $N = 1000$, and therefore, on this example, the Malliavin calculus does not provide any advantage.

We finally report the estimated American option values on five real stocks, whose characteristics are reported in Table 2. The data were obtained from Quandl, now replaced by Data Nasdaq Link (https://data.nasdaq.com/). We first compute the option value for $N \in \{500, 1000, 5000, 10000, 20000\}$ and $M \in \{10, 20, 50, 100, 200, 500\}$, using the LS, RF, and NN approaches. We do not report results with the Malliavin calculus due to the required computational time to produce them. From Fig. 7, showing all the options except GOOG due to scale difference, we see that pure LS method delivers better results than RF, which itself performs better than NN. However, all the methods provides similar results, close to the true option price,

Table 2 Real data characteristics

Stock	V_0	S_0	K	T	r	σ	Trade Date	Maturity	Name
AAPL	8.65	113.700	100.0	1.17808	0.0086	0.310016	2015-11-17	2017-01-20	Apple
BIIB	13.00	279.095	200.0	1.17808	0.0086	0.412166	2015-11-17	2017-01-20	Biogen
GOOG	135.70	726.160	860.0	0.06575	0.0007	0.410721	2015-11-17	2015-12-11	Google
MSFT	17.75	53.020	80.0	0.41096	0.0034	0.424071	2015-11-17	2016-04-15	Microsoft
SBUX	10.25	60.620	70.5	0.06575	0.0007	0.462461	2015-11-17	2015-12-11	Starbucks

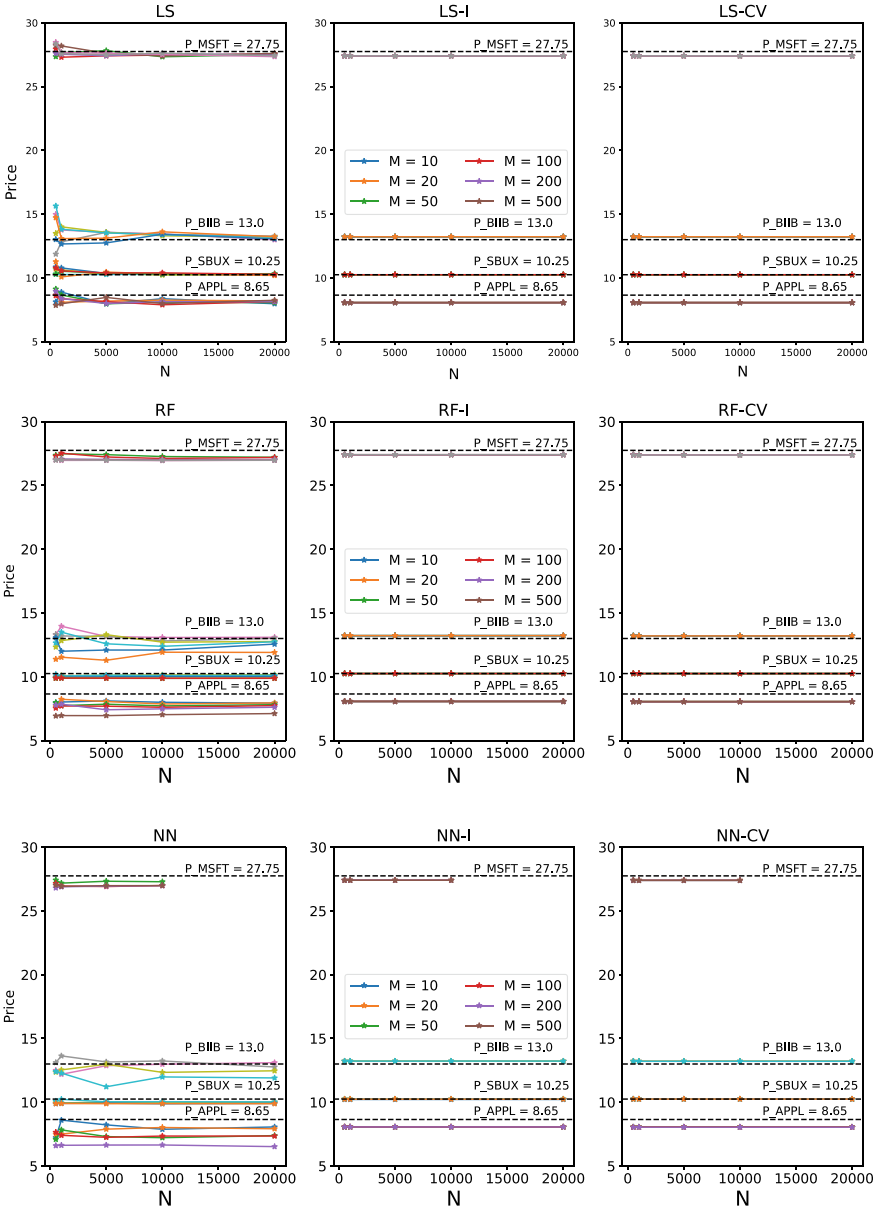


Fig. 7 Real option data comparison

Table 3 Estimated option prices, $N = 5000$, $M = 100$

Stock	LS	LS-I	LS-CV	RF	RF-I	RF-CV
APPL	8.14 (11.69)	8.07 (0.08)	8.07 (0.04)	7.88 (11.47)	8.07 (0.08)	8.06 (0.03)
BIIB	13.53 (26.15)	13.23 (0.11)	13.23 (0.05)	13.63 (24.87)	13.23 (0.11)	13.21 (0.04)
GOOG	136.33 (48.00)	135.72 (0.01)	135.72 (0.01)	134.02 (16.84)	135.72 (0.01)	135.71 (0.01)
MSFT	27.61 (8.91)	27.41 (0.02)	27.41 (0.01)	27.09 (4.36)	27.41 (0.02)	27.39 (0.01)
SBUX	10.46 (5.90)	10.25 (0.00)	10.25 (0.00)	10.01 (2.34)	10.25 (0.00)	10.25 (0.00)

as soon as indirect estimation or control variates are used, even for small values of M and N. The time required to train the neural networks prevented us to report the MISFT value for N greater than 10000 and M greater than 100. In order to better assess the effect of variance reduction, we finally report in Table 3 the option values averaged over 5000 simulated paths, along with the standard deviations in brackets, and $M = 100$. Indirect estimation and control variates again dramatically reduce the standard deviations.

7 Conclusion

American options valuation has been an active research topic for more than three decades, and many approaches have been proposed to tackle it. We have reviewed and compared a few of them, from the least-squares Monte Carlo method [33] to Malliavin calculus and machine learning techniques, on simple examples. While providing accurate results, Malliavin calculus exhibited a large computational burden in addition to the theoretical difficulties to derive the weights, and for practical purposes, it is outperformed by traditional regression techniques. Advanced machine learning techniques, such as neural networks, require careful designs and more research is needed to validate and automatically select them. In particular, we were not able to obtain satisfying results with multilayer perceptions, while random forests deserve more investigation as well selected structures brought promising results. In both cases, we must face higher calibration costs, requiring a large number of scenarios, but option value can also be estimated during this phase, and the computational effort remains reasonable for random forests. Least-squares Monte Carlo approaches performed remarkably well, but we must keep in mind that we reported results on simple examples only. Variance reduction techniques, illustrated by the control variates proposed by Rasmussen [39], appeared to be a crucial ingredient in any of the investigated methods, dramatically improving the results, at a negligible cost.

We nevertheless must remain cautious as more experiments should be performed on more complex situations, including high-dimensional options and value prices with stochastic volatility of jumps. While the Greeks can be estimated by finite difference, specific techniques have been proposed and could also be considered [26].

Some authors [4] suggest that deep learning approaches help to avoid the curse of dimension. However, it is often difficult to collect enough data to accurately calibrate such models and simulation, along with simplifying assumptions on the assets prices processes, remains a key tool. Carefully designed variance reduction techniques allow the estimation of the option price with a limited number of simulated assets price trajectories, and classical parametric regression techniques appear more robust than complex machine learning models that often require a large number of simulations to calibrate, and can underperform if their design is not well selected. We have focused on control variates, but other variance reduction methods, as scenario bundles [26], multilevel Monte Carlo [5], or quasi-Monte Carlo sampling [17], should be also considered, especially as such techniques can be combined [31].

References

1. Alòs, E., Lorite, D.G.: Malliavin Calculus in Finance: Theory and Practice. Chapman and Hall/CRC, Boca Raton, FL, USA (2021)
2. Bally, V., Caramellino, L., Zanette, A.: Pricing and hedging American options by Monte Carlo methods using a Malliavin calculus approach. Technical Report 4804, INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex, France (2004)
3. Bally, V., Caramellino, L., Zanette, A.: Pricing and hedging American options by Monte Carlo methods using a Malliavin calculus approach. Monte Carlo Methods Appl. **11**(2), 97–133 (2005)
4. Becker, S., Cheridito, P., Jentzen, A.: Deep optimal stopping. J. Mach. Learn. Res. **20**(74), 1–25 (2019)
5. Belomestny, D., Dickmann, F., Nagapetyan, T.: Pricing Bermudan options via multilevel approximation methods. SIAM J. Financ. Math. **6**(1), 448–466 (2015)
6. Ben-Ameur, H., L'Ecuyer, P., Lemieux, C.: Combination of general antithetic transformations and control variables. Math. Oper. Res. **29**(4), 946–960 (2004)
7. Blackman, D., Vigna, S.: Scrambled linear pseudorandom number generators. ACM Trans. Math. Softw. **47**(4), 1–32 (2021)
8. Bolia, N., Juneja, S.: Function-approximation-based perfect control variates for pricing American options. In: Kuhl, M.E., Steiger, N.M., Armstrong, F.B., Joines, J.A. (eds.) Proceedings of the 2005 Winter Simulation Conference, pp. 1876–1883 (2005)
9. Bouchard, B., Warin, X.: Monte-Carlo valuation of American options: facts and new algorithms to improve existing methods. In: Carmona, R.A., Del Moral, P., Hu, P., Oudjane, N. (eds.) Numerical Methods in Finance, pp. 215–255. Springer, Berlin Heidelberg, Berlin, Heidelberg (2012)
10. Boyle, P.P.: Options: a Monte Carlo approach. J. Financ. Econ. **4**(3), 323–338 (1977)
11. Broadie, M., Detemple, J.: American option valuation: new bounds, approximations, and a comparison of existing methods. Rev. Financ. Stud. **9**(4), 1211–1250 (1996)
12. Broadie, M., Glasserman, P.: A stochastic mesh method for pricing high-dimensional American options. J. Comput. Financ. **7**(4), 35–72 (2004)
13. Caramellino, L., Zanette, A.: Monte Carlo methods for pricing and hedging American options in high dimension. Risk Decis. Anal. **2**(4), 207–220 (2011)
14. Chan, J.H., Joshi, M., Tang, R., Yang, C.: Trinomial or binomial: accelerating American put option price on trees. J. Futur. Mark. **29**(9), 826–839 (2009)
15. Clément, E., Lamberton, D., Protter, P.: An analysis of a least squares regression method for American option pricing. Financ. Stochast. **6**, 449–471 (2002)

16. Cox, J.C., Ross, S.A., Rubinstein, M.: Option pricing: a simplified approach. J. Financ. Econ. **7**(3), 229–264 (1979)
17. Dion, M., L'Ecuyer, P.: American option pricing with randomized quasi-Monte Carlo simulations. In: Johansson, B., Jain, S., Montoya-Torres, J., Hugan, J., Yücesan, E. (eds.) Proceedings of the 2010 Winter Simulation Conference, pp. 2705–2720. IEEE, Baltimore, MD, USA (2010)
18. Ehrlichman, S.M.T., Henderson, S.G.: Adaptive control variates for pricing multi-dimensional American options. J. Comput. Financ. **11**(1), 65–91 (2007)
19. Fournié, E., Lasry, J.M., Lebuchoux, J., Lions, P.L.: Applications of Malliavin calculus to Monte Carlo methods in finance II. Financ. Stochast. **5**(2), 201–236 (2001)
20. Giles, M.B.: Multilevel Monte Carlo methods. Acta Numer. **24**, 259–328 (2015)
21. Glasserman, P.: Monte Carlo Methods in Financial Engineering. Springer, New York, NY, USA (2004)
22. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, New York, NY, USA (2009)
23. Higham, D.J.: An Introduction to Financial Option Valuation. Cambridge University Press, Cambridge, United Kingdom (2004)
24. Ho, T.K.: Random decision forests. In: Proceedings of the Third International Conference on Document Analysis and Recognition, ICDAR '95, pp. 278–282. IEEE Computer Society, Montreal, QC, Canada (1995)
25. Hull, J., White, A.: The use of the control variate technique in option pricing. J. Financ. Quant. Anal. **23**(3), 237–251 (1988)
26. Jain, S., Oosterlee, C.W.: The stochastic grid bundling method: efficient pricing of Bermudan options and their Greeks. Appl. Math. Comput. **269**, 412–431 (2015)
27. James, G., Witten, D., Hastie, T., Tibshirani, R.: An Introduction to Statistical Learning: With Applications in R, 2nd edn. Springer, New York, NY, USA (2021)
28. Kharrat, M., Bastin, F.: Continuation value computation using Malliavin calculus under general volatility stochastic process for American option pricing. Turk. J. Math. (2021)
29. L'Ecuyer, P.: Good parameters and implementations for combined multiple recursive random number generators. Oper. Res. **47**(1), 159–164 (1999)
30. L'Ecuyer, P.: Variance reduction's greatest hits. In: Proceedings of the 2007 European Simulation and Modeling Conference, pp. 5–12. EUROSIS, Ghent, Belgium (2007)
31. Lemieux, C., La, J.: A study of variance reduction techniques for American option pricing. In: Kuhl, M.E., Steiger, N.M., Armstrong, F.B., Joines, J.A. (eds.) Proceedings of the 2005 Winter Simulation Conference, pp. 1884–1891. IEEE, Orlando, Florida (2005)
32. Lions, P.L., Régnier, H.: Calcul du prix et des sensibilités d'une option américaine par une méthode de Monte Carlo. Technical report, Ceremade, Paris, France (2001)
33. Longstaff, F.A., Schwartz, E.S.: Valuing American options by simulation: a simple least-squares approach. Rev. Financ. Stud. **14**(1), 113–147 (2001)
34. Malliavin, P.: Stochastic calculus of variations and hypoelliptic operators. In: Proceedings of the International Symposium on Stochastic Differential Equations, Kyoto, 1976, pp. 195–263. Wiley, New York, NY, USA (1978)
35. Malliavin, P., Thalmaier, A.: Stochastic calculus of variations in mathematical finance. Springer, Berlin, Germany (2006)
36. Merton, R.C.: Theory of rational option pricing. Bell J. Econ. Manag. Sci. **4**(1), 141–183 (1973)
37. Pascucci, A.: PDE and Martingale Methods in Option Pricing. Springer, Milan, Italy (2010)
38. Rabia, M.: Numerical methods for high dimensional backward stochastic differential equations. Master's thesis, National University of Singapore, Singapore (2017)
39. Rasmussen, N.S.: Control variates for Monte Carlo valuation of American options. J. Comput. Financ. **9**(1), 83–118 (2005)
40. Rendleman, R.J., Bartter, B.J.: Two-state option pricing. J. Financ. **34**(5), 1093–1110 (1979)
41. Ruf, J., Wang, W.: Neural networks for option pricing and hedging: a literature review. J. Comput. Financ. **24**(1), 1–46 (2020)
42. West, L.: American Monte Carlo option pricing under pure jump Lévy models. Master's thesis, Stellenbosch University, Stellenbosch, Western Cape, South Africa (2013)
43. Wu, Z.: Pricing American options using Monte Carlo method. Master's thesis, University of Oxford, Oxford, United Kingdom (2012)

Remarks on Lévy Process Simulation

Søren Asmussen

Abstract Algorithms for simulation of a Lévy process $X(t)$ are discussed, with particular emphasis on two algorithms approximating jumps that are in some sense small. One is classical, defining small jumps as those of absolute value $<\varepsilon$. The other one appears to be new and relies on an completely monotone structure of the Lévy density $n(x)$. One then truncates the representing measure of $n(x)$ to $[0, A]$, meaning that jumps of mean $<1/A$ are left out. In both algorithms, the large jump part is simulated as compound Poisson and the small jumps are approximated. The standard choice of such an approximation is normal with the same mean and variance, but we also consider gamma approximations in two variants, and show that in some cases these perform substantially better. Other algorithms are briefly surveyed and we sketch a new one for simulation of a tempered stable (CGMY) process with infinite variation.

Keywords Acceptance-rejection · Complete monotonicity · Conditional Monte Carlo · Lévy measure · Tempered stable process

1 Introduction

A Lévy process $X(t)$ has the structure $X(t) = at + \sigma W(t) + J(t)$ where $W(t)$ is standard Brownian motion (BM) and $J(t)$ an independent pure jump process (see further below). This class of processes has been used in numerous application areas, of which we in particular mention finance [14, 35] and queueing [15].

Calculations for a Lévy process are, however, in general more difficult than for BM, and an abundance of expressions that are explicit for BM are not so even in the most popular parametric Lévy models. Simulation of $X(t)$ is therefore one of the main computational tools. For example in finance, it is most often the simplest vehicle for evaluating option prices of the form

S. Asmussen (✉)
Department of Mathematics, Aarhus University, Ny Munkegade, 8000 Aarhus C, Denmark
e-mail: bastin@iro.umontreal.ca

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_2

$$\mathbb{E}\Psi\big(X(0:T)\big) \tag{1}$$

where T is the maturity time, $X(0:T)$ stands for the whole path $\big\{X(t)\big\}_{0\le t\le T}$, and Ψ is a suitable path functional.

The simulation of the $at+\sigma W(t)$ component is straightforward, so we assume that $X(t)$ is a pure jump process. The main characteristic of such a process is its Lévy measure ν, which with a few exceptions we throughout assume absolutely continuous with density $n(x)\ge 0$. Conditions needed on ν are $\int_{|x|>\varepsilon}\nu(\mathrm{d}x)<\infty$ and $\int_{|x|\le\varepsilon}x^2\nu(\mathrm{d}x)<\infty$ for some (and then all) $\varepsilon>0$. The process is said to have finite activity if $\lambda=\int_{\mathbb{R}}\nu(\mathrm{d}x)<\infty$ and is then a compound Poisson process with Poisson rate λ and density $n(x)/\lambda$ of the jumps. Sample paths of $X(t)$ are of finite variation if and only if $\int_{|x|\le\varepsilon}x\nu(\mathrm{d}x)<\infty$. The picture is roughly that jumps in $[x,x+\mathrm{d}x)$ occur at Poisson rate $n(x)\,\mathrm{d}x$ and independently for different values of x. In the infinite variation case, $X(t)$ is, however, only completely specified by $n(x)$ up to a drift term (see further Sect. 2). The process is called spectrally positive or negative if $n(x)\equiv 0$ for $x<0$, resp. $x>0$; otherwise, we refer to it as two-sided. In finance, the most popular classes of jump processes are the NIG (Normal Inverse Gaussian), tempered stable (TS or CGMY), VG (Variance Gamma) and Meixner ones, and we survey these in Sect. 3.

Exact simulation of the whole path $X(0:T)$ is obviously impossible due to the presence of infinitely many jumps of the process. One could hope that one can perform exact simulation of $X(T)$ for any given T and thereby a discrete skeleton $X(h), X(2h),\dots$ for any h. As surveyed briefly in Sect. 8, this is simple for VG, with a little added effort also possible for NIG and CGMY with finite variation, and presumably possible but quite tedious for Meixner. In general, this is however not feasible and we focus on two approximative alternatives. They both consist in simulating the finite number of jumps which are in some sense "big" as a compound Poisson process, and replacing the infinity of the remaining "small" ones with an easily simulated approximation. The path $X(0:T)$ can then by obtaining by assigning i.i.d. uniform $[0,T]$ location to the jumps and possibly filling in some information provided by the particular form of the approximation. The first of these approaches is classical and widely applied, and simply defines the big jumps as those of absolute value $>\varepsilon$; we refer to this as the ε-algorithm. These jumps are those coming from the part of ν concentrated on $\{|x|>\varepsilon\}$. By definition, this is a finite measure and so the corresponding contribution to X can be simulated as a compound Poisson process. The second approach, which does not appear to have been considered in the simulation literature, relies on a completely monotone (CM) structure

$$n(x) \;=\; \int_0^\infty \mathrm{e}^{-xt}\,V(\mathrm{d}t) \;=\; \int_0^\infty \mathrm{e}^{-xt}v(t)\,\mathrm{d}t \tag{2}$$

of the Lévy density where V is a Radon measure with density v. This holds in many main examples and represents the jumps as an infinite mixture of exponential(t) jumps with the rate t having weight $v(t)/t$ (see further Sect. 5). The compound

Poisson part is then obtained by restricting V to $(0, A)$ for some $A < \infty$, meaning that exponential jumps with mean $< 1/A$ are left out. We refer to the method as the CM-algorithm. In both approaches, the computational effort as measured by the Poisson mean goes to infinity as $\varepsilon \to 0$, resp. $A \to \infty$. As for the approximation of the small jump part, the standard choice in the ε-algorithm is a normal with the same mean and variance and is substantiated in [6] by a limit result as $\varepsilon \to 0$ (further relevant references pertaining to this are [16, 37]). However, we shall also consider gamma alternatives in 2–3 variants and illustrate by examples that these perform at least as well, in some cases even convincingly better. Doing so, our point of view is largely empirical: for the practitioner, comparison of approaches as $\varepsilon \to 0$ matters less than performance for ε so moderate that the computational effort is within reach. As $\varepsilon \to 0$, the small jumps contribute less, and hence limit results become less relevant. Similar remarks apply to the CM-algorithm. We also point out that in some types of applications, the approximation of the small jumps need not necessarily be simulated, but instead it may be used via conditional Monte Carlo for providing smooth density estimates and variance reduction.

2 Lévy Processes

For the general theory of Lévy processes, see e.g. [33] and [10]. A jump process is constructed from a Poisson random measure $L(\mathrm{d}t, \mathrm{d}x)$ on $(0, \infty) \times \mathbb{R}/\{0\}$ with intensity measure $\mathrm{d}t \otimes \nu(\mathrm{d}x)$. In the finite variation case $\int |x|\, \nu(\mathrm{d}x) < \infty$, one has

$$X(t) \;=\; \int_{s \le t,\, x \in \mathbb{R}} x\, L(\mathrm{d}s, \mathrm{d}x)\,, \quad \kappa(\theta) \;=\; \int_{-\infty}^{\infty} \left(\mathrm{e}^{\theta x} - 1\right) \nu(\mathrm{d}x) \tag{3}$$

at least for $\Re(\theta) = 0$ and in our examples in a strip containing the imaginary axis. Here $\kappa(\theta) = \log \mathbb{E}\mathrm{e}^{\theta X(1)}$ is the so-called Lévy exponent or cumulant function. In the infinite variation case, there are too many small jumps for these integrals to converge. Instead, so-called compensation is needed and consists in appropriate centerings and limits. Traditionally, jumps of absolute size < 1 are centered, which leads to

$$X(t) \;=\; at + \lim_{\varepsilon \to 0} \left\{ \int_{s \le t,\, \varepsilon < |x| < \infty} x\, L(\mathrm{d}s, \mathrm{d}x) - t \int_{\varepsilon < |x| \le 1} x\, \nu(\mathrm{d}x) \right\}, \tag{4a}$$

$$\kappa(\theta) \;=\; a + \int_{-\infty}^{\infty} \left(\mathrm{e}^{\theta x} - 1 - \theta x \mathbb{I}(|x| \le 1)\right) \nu(\mathrm{d}x) \tag{4b}$$

for some a. Obviously, taking 1 as truncation point is arbitrary, and other choices lead to different values of a. If the mean $\mathbb{E}X(1) = \kappa'(0)$ is finite, it may be more convenient to center all jumps, and one then has

$$X(t) \;=\; t\kappa'(0) + \lim_{\varepsilon\to 0}\left\{\int_{s\le t,\,|x|>\varepsilon} x\,L(\mathrm{d}s,\mathrm{d}x) - t\int_{|x|>\varepsilon} x\,\nu(\mathrm{d}x)\right\}, \tag{5a}$$

$$\kappa(\theta) \;=\; \kappa'(0) + \int_{-\infty}^{\infty}\left(\mathrm{e}^{\theta x} - 1 - \theta x\right)\nu(\mathrm{d}x)\,. \tag{5b}$$

The cumulants κ_k of $X(1)$ are given as the kth derivatives $\kappa^{(k)}(0)$ of $\kappa(\theta)$ at $\theta = 0$. In particular, $\kappa_1 = \mathbb{E}X(1)$, $\kappa_2 = \mathbb{V}arX(1)$, and the skewness and (excess) kurtosis are $\kappa_3/\kappa_2^{3/2}$, resp. κ_4/κ_2^2. For $k \ge 2$, one alternatively has

$$\kappa_k \;=\; \int_{-\infty}^{\infty} x^k\,\nu(\mathrm{d}x), \tag{6}$$

and this expression is also valid for $k = 1$ in the finite variation case.

3 Main Examples

In the absolutely continuous case, define the Lévy density $n(x) = \mathrm{d}\nu(x)/\mathrm{d}x$ as the density of the Lévy measure w.r.t. Lebesgue measure.

The NIG process [9] has parameters $\alpha, \delta > 0$, $\beta \in (-\alpha, \alpha)$ and $\mu \in \mathbb{R}$. The Lévy density is

$$n(x) \;=\; \frac{\alpha\delta}{\pi|x|}K_1\big(\alpha|x|\big)\mathrm{e}^{\beta x}, \quad x \in \mathbb{R}, \tag{7}$$

where as usual $K_1(z)$ denotes the modified Bessel function of the third kind with index 1. The cumulant function and the density of $X(1)$ are, respectively,

$$\kappa(s) \;=\; \mu s + \delta\left(\sqrt{\alpha^2-\beta^2} - \sqrt{\alpha^2-(\beta+s)^2}\right),\ \alpha-\beta < \Re(s) < \alpha+\beta\,,$$

$$\frac{\alpha\delta}{\pi}\exp\big\{\delta\sqrt{\alpha^2-\beta^2} + \beta(x-\mu)\big\}\frac{K_1\big(\alpha\sqrt{\delta^2+(x-\mu)^2}\big)}{\sqrt{\delta^2+(x-\mu)^2}}\,.$$

The Meixner (MX) process [18, 28, 35] has parameters $a, d > 0$, $b \in (-\pi, \pi)$ and $m \in \mathbb{R}$. The Lévy density is

$$n(x) \;=\; d\frac{\exp\{bx/a\}}{x\sinh(\pi|x|/a)} \;=\; 2d\frac{\exp\big\{bx/a - \pi|x|/a\big\}}{|x|\big(1-\exp\{-2\pi|x|/a\}\big)}\,. \tag{8}$$

The cumulant function and the density of $X(1)$ are, respectively,

$$\kappa(s) \;=\; 2d\log\bigl(\cos(b/2)\bigr) - 2d\log\bigl(\cos(as+b)/2)\bigr) + ms\,, \quad \tfrac{\pi+b}{a} < \Re(s) < \tfrac{\pi-b}{a}\,,$$
$$\frac{(2\cos(b/2))^{2d}}{3a\pi\,\Gamma(2d)}\mathrm{e}^{b(x-m)/a}\bigl|\Gamma(d+\mathrm{i}(x-m)/a)\bigr|^2\,. \tag{9}$$

For the tempered stable (TS) process [3, 12, 24]

$$n(x) \;=\; \delta_{\pm}\mathrm{e}^{-\beta_{\pm}|x|}/|x|^{\alpha_{\pm}+1} \tag{10}$$

where δ_+, β_+ are for $x > 0$ and δ_-, β_- for $x < 0$. When $\delta_+ = \delta_-, \alpha_+ = \alpha_-$, the TS process goes under the acronym CGMY process in particular in finance, where the traditional notation is $\delta_+ = \delta_- = C, \alpha_+ = \alpha_- = Y, G$ instead of β_- and M instead of β_+. Cf. the author names in [12]! In terms of the positive jumps, $\alpha_+ < 0$ corresponds to a compound process, $\alpha_+ = 0$ to a gamma process where $X(1)$ is gamma distributed with shape parameter δ_+ and rate parameter β_+, $0 < \alpha_+ < 1$ to infinite activity but finite variation, and $1 \le \alpha_+ < 2$ to infinite variation. The cumulant function is

$$\kappa(s) \;=\; \delta_-\Gamma(-\alpha_-)\bigl((\beta_- + s)^{\alpha_-} - \beta_-^{\alpha_-}\bigr) + \delta_+\Gamma(-\alpha_+)\bigl((\beta_+ - s)^{\alpha_+} - \beta_+^{\alpha_+}\bigr)\,, \tag{11}$$

$-\beta_- < \Re(s) < \beta_+$. Here and at other places in the theory, exceptions apply when α_+ or α_- or both equals 0 or 1. The case $\alpha_+ = \alpha_- = 0$ is the VG process (the difference between two gamma processes).

Starting from [12, 13], the density of $X(1)$ in the TS process has traditionally been computed by Fourier inversion via (11). However, it is pointed out in [3] that the density can be expressed as

$$f(x) \;=\; \exp\{-\beta x - \delta\Gamma(-\alpha)\beta^{\alpha}\}\, f_0(x) \tag{12}$$

where f_0 is the density of a strictly α-stable distribution $S_\alpha(\sigma, 1, 0)$ distribution with $\sigma = \bigl(-\delta\Gamma(-\alpha)\cos(\pi\alpha/2)\bigr)^{1/\alpha}$. See also [27, 30]. Given the availability of software for stable distributions, (12) provides an easy approach to numerical computations.

In all these examples, one has

$$n(x) \;\sim\; \frac{\delta}{x^{1+\alpha^*}} \quad \text{as } x \downarrow 0 \tag{13}$$

for some δ and some $\alpha^* \in [0, 2)$ (subject to this, α^* is sometimes referred to as the Blumenthal-Getoor index). In fact, for TS this holds since $\mathrm{e}^{-\beta x} \to 1$, whereas one has $\alpha^* = 1$ for NIG and MX, as follows from known asymptotics of K_1, resp. $1 - \exp\{-2\pi x a\} \sim 2\pi x/a$.

4 The ε–Algorithm

Typically, the positive and negative jumps are simulated separately, so we consider only the spectrally positive case in the following.

When truncating the jumps to $[\varepsilon, \infty)$, the exactly simulated compound Poisson part of $X(1)$ is $X_{\varepsilon,\infty}(1) = \sum_1^N Y_n(\varepsilon)$ where N is Poisson $\lambda(\varepsilon)$ and $Y_1(\varepsilon), Y_2(\varepsilon), \ldots$ are i.i.d. with density $g(x;\varepsilon)$ with

$$\lambda(\varepsilon) = \int_\varepsilon^\infty n(x)\,\mathrm{d}x\,, \quad g(x;\varepsilon) = \frac{n(x)}{\lambda(\varepsilon)}\,, \ \varepsilon < x < \infty.$$

Some approximation $\widehat{X}_{0,\varepsilon}(1)$ of jumps of value $< \varepsilon$ is then used, and one returns the r.v. $\widehat{X}_{0,\varepsilon}(1) + X_{\varepsilon,\infty}(1)$. For these approximations, one typically needs the cumulants of $X_{0,\varepsilon}(1)$ which according to (6) are $\kappa_{k;0,\varepsilon} = \int_0^\varepsilon x^k\,\nu(\mathrm{d}x)$ if either $k \ge 2$ or $k \ge 1$ and the process has finite variation; in the infinite variation case, $\kappa_{1;0,\varepsilon} = 0$ subject to (5a). In practice, $\int_0^\varepsilon x^k\,\nu(\mathrm{d}x)$ is seldom explicit, but needs to be evaluated by numerical integration. Alternatively, one may note that subject to (13), one has

$$\kappa_{k;0,\varepsilon} \;=\; \int_0^\varepsilon x^k\,\nu(\mathrm{d}x) \;\sim\; \delta\frac{\varepsilon^{k-\alpha^*}}{k-\alpha^*} \quad \text{if } \alpha^* < 1,\ k \ge 1 \text{ or } 1 \le \alpha^* < 2,\ k \ge 2. \tag{14}$$

The most naive choice is $\widehat{X}_{0,\varepsilon}(1) \equiv 0$. However, it was suggested in [11] and [32] to take $X_{0,\varepsilon}(t)$ as a BM with fitted mean and variance when $\varepsilon < 1$. Supporting limit theorems were given in [6], establishing the validity of this procedure when X is not too close to the finite activity case $\int \nu(\mathrm{d}x) < \infty$ and ν satisfies some weak smoothness conditions (a simple proof under the stronger condition (13) follows by paralleling the proof of Proposition 3 below). We shall here suggest gamma alternatives in two variants.

Recall that the gamma distribution with shape parameter r and rate parameter b has density $b^r x^{r-1}\mathrm{e}^{-bx}/\Gamma(r)$ and cumulant function $\log\bigl(b/(b-z)\bigr)^r = -r\log(1-z/b)$ with kth derivative $r(k-1)!b^{-k}(1-z/b)^{-k}$. Thus the kth cumulant is $\kappa_k = r(k-1)!/b^k$; in particular the skewness is $(2r/b^3)/(r/b^2)^{3/2} = 2r^{-1/2}$. Given a distribution or a set of data with cumulants $\kappa_k^\#$, the most obvious possibility is to fit the mean and variance which leads to

$$b = \frac{\kappa_1^\#}{\kappa_2^\#}, \quad r \;=\; b\kappa_1^\# \;=\; \frac{\kappa_1^{\#2}}{\kappa_2^\#}\,. \tag{Γ_1}$$

One could also consider a three-parameter gamma family by allowing a shift m, and fitting the mean, variance and skewness then gives

$$r \;=\; \frac{4\kappa_2^{\#3}}{\kappa_3^{\#2}}\,, \quad b = \sqrt{\frac{r}{\kappa_2^\#}} \;=\; \frac{2\kappa_2^\#}{\kappa_3^\#}\,, \quad m \;=\; \kappa_1^\# - \frac{r}{b}. \tag{Γ_2}$$

Note that for a Lévy process, (Γ_1) does not make sense in the infinite variation case since then $\kappa_{1;0,\varepsilon} = 0$ subject to (5). For a subordinator (a spectrally positive process with a non-negative linear drift), (Γ_2) may be controversial because it may destroy the property of the process being non-decreasing. The normal approximation has the same problem, but not (Γ_1). Both of (Γ_1), (Γ_2) asymptotically agree with the normal approximation as $\varepsilon \downarrow 0$. This follows since (14) implies that $b \to \infty$ in both cases, which implies a gamma distribution to be asymptotically normal.

Efficiently generating r.v.'s from the density $g(x;\varepsilon) = n(x)/\lambda(\varepsilon)$, $x > \varepsilon$ may not always be trivial. However, a general set-up covering many examples is

$$(n_1 n_2) \quad n(x) = \frac{n_1(x)}{n_2(x)} \text{ for } x > 0 \text{ with } n_1(x) \text{ strictly decreasing, } n_2'(x) > 0,$$
$$n_1(x) \text{ integrable on } (x_0, \infty) \text{ and } 1/n_2(x) \text{ on } (\varepsilon, x_0) \text{ for all } 0 < \varepsilon < x_0 < \infty.$$

In the TS situation, $n_1(x) = d\mathrm{e}^{-\beta x}$, $n_2(x) = x^{1+\alpha}$; for the positive jumps of MX, one may take $n_1(x) = 2d\exp\{bx/a - \pi x/a\}$, $n_2(x) = x\big(1 - \exp\{-2\pi x/a\}\big)$; etc.

Even for the TS case, the c.d.f. of $g(x;\varepsilon)$ is not explicitly available. Thus inversion is not feasible and acceptance-rejection (A-R) seems the reasonable approach. What suggests itself is to either use the exponential(β) distribution on (ε, ∞) as proposal and reject w.p. proportional to $1/x^{1+\alpha}$, or to use the Pareto(α) distribution on (ε, ∞) as proposal and reject w.p. proportional to $\mathrm{e}^{-\beta x}$. However, the first procedure would lead to a high rejection rate for small or moderate x, and the second for large or moderate x. So, a reasonable compromise is to choose some threshold x_0 and use the Pareto proposal on (ε, x_0) and the exponential on (x_0, ∞). An equivalent formulation is to decompose $X_{\varepsilon,\infty}$ into two compound Poisson terms, one having jumps in $(\varepsilon, x_0]$ and the other having jumps in (x_0, ∞). Note that the proposal on (ε, x_0) (a truncated Pareto) is easily simulated by inversion as $\big(1/\varepsilon^\alpha - \alpha\mu_2(x_0)U\big)^{-1/\alpha}$ with U uniform(0, 1), cf. [4, p. 39].

In order to analyze this A-R procedure in the general set-up of $(n_1 n_2)$, define for a fixed $\varepsilon > 0$

$$\lambda_1(x_0) = \int_\varepsilon^{x_0} n(x)\,\mathrm{d}x\,, \quad \mu_1(x_0) = \int_\varepsilon^{x_0} \frac{1}{n_2(x)}\,\mathrm{d}x\,, \quad C_1(x_0) = \frac{n_1(\varepsilon)\mu_1(x_0)}{\lambda_1(x_0)}\,,$$
$$\lambda_2(x_0) = \int_{x_0}^{\infty} n(x)\,\mathrm{d}x\,, \quad \mu_2(x_0) = \int_{x_0}^{\infty} n_1(x)\,\mathrm{d}x\,, \quad C_2(x_0) = \frac{\mu_2(x_0)}{\lambda_2(x_0)n_2(x_0)}\,.$$

The target distributions are then

$$f_1(x) = \frac{n_1(x)}{\lambda_1(x_0)n_2(x)},\ \varepsilon < x < x_0, \quad \text{and} \quad f_2(x) = \frac{n_1(x)}{\lambda_2(x_0)n_2(x)},\ x_0 < x < \infty,$$

and the proposals are

$$g_1(x) = \frac{1}{\mu_1(x_0)n_2(x)},\ \varepsilon < x < x_0, \quad \text{and} \quad g_2(x) = \frac{n_1(x)}{\mu_2(x_0)},\ x_0 < x < \infty.$$

Then $f_1(x) \le C_1(x_0)g_1(x)$ and $f_2(x) \le C_2(x_0)g_2(x)$, and we may use A-R with acceptance probabilities

$$\frac{f_1(x)}{C_1(x_0)g_1(x)} = \frac{n_1(x)\mu_1(x_0)}{\lambda_1(x_0)C_1(x_0)}, \quad \frac{f_2(x)}{C_2(x_0)g_2(x)} = \frac{\mu_2(x_0)}{\lambda_2(x_0)C_2(x_0)n_2(x)}$$

for r.v. generation from f_1, resp. f_2. This gives expected numbers $C_1(x_0)$, $C_2(x_0)$ of samplings from $g_1(x)$, resp. $g_2(x)$, and as measure $E(x_0)$ of the computational effort, we shall use the total number of these samplings, i.e.

$$E(x_0) \;=\; \lambda_1(x_0)C_1(x_0) + \lambda_2(x_0)C_2(x_0) \;=\; n_1(\varepsilon)\mu_1(x_0) + \frac{\mu_2(x_0)}{n_2(x_0)}\,.$$

Of course, if the costs to generate from $g_1(x)$, resp. $g_2(x)$ are very different, $E(x_0)$ needs to be reflected to reflect this disparity.

Proposition 1 *Consider the function $E(x_0)$, $\varepsilon \le x_0 \le \infty$, If $n_2'(x_0)/n_2(x_0) \to 0$ as $x_0 \to \infty$, then $E(x_0)$ attains its minimum for some $\varepsilon < x_0^* < \infty$ satisfying $\psi(x_0^*) = 0$ where $\psi(x_0) = n_2(x_0)\big(n_1(\varepsilon) - n_1(x_0)\big) - \mu_2(x_0)n_2'(x_0)$. In particular, for the TS case x_0^* is the unique solution in (ε, ∞) of*

$$x_0^*(\mathrm{e}^{\beta(x_0^*-\varepsilon)} - 1) \;=\; \frac{1+\alpha}{\beta}. \tag{15}$$

Proof We have $\frac{\mathrm{d}}{\mathrm{d}x_0}E(x_0) = \psi(x_0)/n_2(x_0)^2$. Here $\psi(\varepsilon) = -\mu_2(\varepsilon)n_2'(\varepsilon) < 0$. As $x_0 \to \infty$, we have $\liminf\big(n_1(\varepsilon) - n_1(x_0)\big) > 0$ and $\mu_2(x_0) \to 0$, and so $n_2'(x_0)/n_2(x_0) \to 0$ implies $\psi(x_0) > 0$ for all large x_0. This gives the first part of the result. For the second on the TS case, we get

$$\psi(x_0) \;=\; x_0^{1+\alpha}(d\mathrm{e}^{-\beta\varepsilon} - d\mathrm{e}^{-\beta x_0}) - d(\mathrm{e}^{-\beta x_0}/\beta)\cdot(1+\alpha)x_0^{\alpha}\,.$$

Multiplying by $\mathrm{e}^{-\beta x_0}$ and rearranging shows that $\psi(x_0^*) = 0$ is the same as (15). For uniqueness of the solution, note that the l.h.s. of (15) is strictly increasing in x_0^* with limits 0 at $x_0^* = \varepsilon$ and ∞ at $x_0^* = \infty$. □

5 Using Complete Monotonicity Structure

Again, we consider only the spectrally positive case and assume the Lévy measure $n(x)$ to be completely monotone in the sense of (2). We refer to the measure $V(\mathrm{d}t)$ as the reference measure and to $v(t)$ as the reference density. See, e.g., [34] for background on complete monotonicity and a huge list of examples. Motivation and financial examples are in [12, 19, 21].

Example 1 We check here that complete monotonicity holds in our main examples. We use the rule that if $m(x)$ is completely monotone with reference density $v(t)$, $t > 0$, then $\mathrm{e}^{-\beta x} m(x)$ is completely monotone with reference density $v(t-\beta)$ for $t > \beta$ and $= 0$ for $0 < t < \beta$.

In the NIG case, this rule together with the standard formula $K_1(x) = x \int_1^\infty \mathrm{e}^{-xt}(t^2-1)^{1/2}\mathrm{d}t$ and elementary substitutions gives the expression

$$v(t) \;=\; \frac{\delta}{\pi}\sqrt{(t+\beta)^2-\alpha^2}\,, \quad t > \alpha - \beta,$$

for the reference density for the positive part of the Lévy measure. For MX, let $\chi(t) = \lceil t \rceil$ be the step function equal to $n+1$ for $t \in (n, n+1]$. Then

$$\begin{aligned}\frac{1}{1-\mathrm{e}^{-x}} \;&=\; 1 + \mathrm{e}^{-x} + \mathrm{e}^{-2x} + \cdots \;=\; 1 - \mathrm{e}^{-x} + 2(\mathrm{e}^{-x} - \mathrm{e}^{-2x}) + 3(\mathrm{e}^{-2x} - \mathrm{e}^{-3x}) + \cdots \\ &=\; \sum_{n=0}^{\infty} (n+1) x \int_n^{n+1} \mathrm{e}^{-xt}\,\mathrm{d}t \;=\; x \int_0^\infty \mathrm{e}^{-xt}\chi(t)\,\mathrm{d}t\end{aligned}$$

which gives

$$v(x) \;=\; 2d\,\chi\big(a(t-\pi/a+b/a)/(2\pi)\big)\,, \quad t > \pi/a - b/a.$$

Finally for the TS case, it is shown in [12] that $v(t) = \delta(t-\beta)^\alpha/\Gamma(1+\alpha)$, $t > \beta$, which in turn is an easy consequence of $\int_0^\infty \mathrm{e}^{-xt} t^\alpha\,\mathrm{d}t = \Gamma(1+\alpha)/x^{1+\alpha}$. ♢

In all three examples, the reference density $v(t)$ grows at rate t^{α^*} as $t \to \infty$, with α^* as in (13). This is in fact no coincidence since Feller's Tauberian theorem [17, p. 445] implies that $V(t) = \int_0^t v(s)\,\mathrm{d}s \sim \delta t^{1+\alpha^*}/\Gamma(2+\alpha^*)$. Hence by formal differentiation,

$$v(t) \;\sim\; \delta(1+\alpha^*)t^{\alpha^*}/\Gamma(2+\alpha^*) \;=\; \delta t^{\alpha^*}/\Gamma(1+\alpha^*). \tag{16}$$

We stress that this is formal: the known rigorous result in this direction requires (beyond existence of v) that v is monotone, cf. [36]. However, we shall take (16) as an assumption for the further developments to follow.

In the following, we use that (2), (6) and Fubini's theorem give the representation

$$\int_0^\infty x^k n(x)\,\mathrm{d}x \;=\; \int_0^\infty \left(\int_0^\infty x^k \mathrm{e}^{-tx}\,\mathrm{d}x\right) v(t)\,\mathrm{d}t \;=\; \int_0^\infty \frac{k!}{t^{k+1}} v(t)\,\mathrm{d}t \tag{17}$$

of the cumulants for $k = 0, 1, \ldots$ As in Sect. 4, we decompose the Lévy density n into two components, here taken as

$$n_{0,A}(x) \;=\; \int_0^A \mathrm{e}^{-xt} v(t)\,\mathrm{d}t\,, \quad n_{A,\infty}(x) \;=\; \int_A^\infty \mathrm{e}^{-xt} v(t)\,\mathrm{d}t\,.$$

The corresponding decomposition of X is written as $X = X_{0,A} + X_{A,\infty}$. The key to our algorithm using complete monotonicity is the following:

Proposition 2 *Assume the measure V in (2) is finite and let*

$$\mu = V(\infty) = \int_0^\infty \frac{v(t)}{t}\,\mathrm{d}t\,, \quad \lambda = \int_0^\infty n(x)\,\mathrm{d}x\,.$$

Then $\mu = \lambda$. Let further T be standard exponential, Y a independent r.v. with density $\frac{v(t)}{t\mu}$ and Z one with density $n(x)/\lambda$. Then $T/Y = Z$ in distribution.

Proof Taking $k = 0$ in (17) gives $\lambda = \mu$. We then get

$$\begin{aligned}\mathbb{P}\big(T/Y \in \mathrm{d}x\big) &= \int_0^\infty \mathbb{P}\big(T/Y \in \mathrm{d}x \,\big|\, Y = t\big)\frac{v(t)}{t\mu}\,\mathrm{d}t \\ &= \int_0^\infty t\mathrm{e}^{-tx}\,\frac{v(t)}{t\mu}\,\mathrm{d}t \;=\; \frac{n(x)}{\mu} \;=\; \mathbb{P}(Z \in \mathrm{d}x)\,.\end{aligned}$$

□

This suggests that in the finite variation case, we can generate a r.v. X approximately distributed as $X(1)$ as follows (more details on the individual steps are given below):

(1) Choose $A < \infty$, let $\lambda = \int_0^A v(t)/t\,\mathrm{d}t$ and generate N as Poisson(λ).
(2) Generate $X_1 = \sum_{n=1}^N T_n/Y_n(A)$ where the T_n are standard exponential and the $Y_n(A)$ have density $v(t)/(\lambda t)$, $0 < t < A$.
(3) Generate X_2 as some approximation to $X_{A,\infty}(1)$.
(4) Return $X = X_1 + X_2$.
In the infinite variation case subject to (5), replace X_1 in (2) by

$$\sum_{n=1}^N \frac{T_n}{Y_n(A)} - \int_0^\infty x n_{0,A}(x)\,\mathrm{d}x \;=\; \sum_{n=1}^N \frac{T_n}{Y_n(A)} - \int_0^A \frac{v(t)}{t^2}\,\mathrm{d}t$$

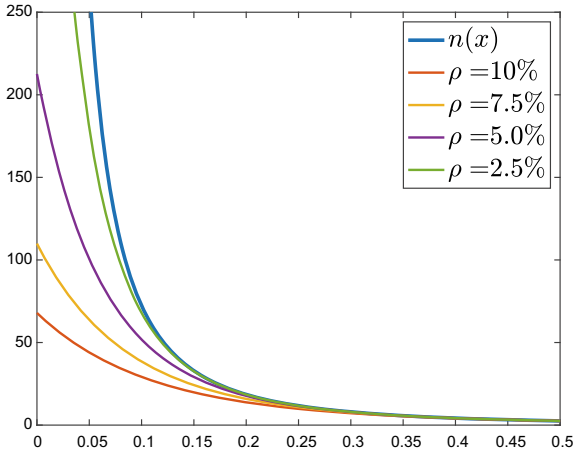
and X in (4) by $\kappa'(0) + X_1 + X_2$. In both cases, $X \to X(1)$ as $A \to \infty$.

That λ in (1) is finite follows by the Radon property of $V(\mathrm{d}x)$. The shape of the part $n_{0,A}$ of n corresponding to the simulated large jumps is illustrated in Fig. 1, The process in the example is TS with $\alpha = 0.8$, variance $\kappa_2 = 1$, kurtosis $K = 2$ and there are 4 values of A determined by the ρ defined as the proportion $\mathbb{V}ar\big(X_{A,\infty}(1)\big)/\kappa_2$ of the total variance provided by the small jumps (see further Sect. 6).

As for the approximation in (3), the most obvious choice is a normal distribution with the correct mean and variance, and this is in fact justified by the following result (recall that W denotes BM):

Proposition 3 *Define $X^*_{A,\infty}(t) = \big(X_{A,\infty}(t) - t\mathbb{E}X_{A,\infty}(1)\big)\big/\sqrt{\mathbb{V}ar\,X_{A,\infty}(1)}$. Then $X^*_{A,\infty} \xrightarrow{\mathcal{D}} W$ in the Skorokhod space $D[0,\infty)$ as $A \to \infty$.*

Fig. 1 $n(x)$ and $n_{0,A}(x)$



Proof Let κ_k^* be the kth cumulant of $X_{A,\infty}^*(1)$. Then κ_k^* is of order $A^{\alpha-k}A^{(2-\alpha)k/2} = A^{\alpha((1-k/2)}$ for $k > 2$ since by (17)

$$\begin{aligned}\frac{\Gamma(1+\alpha)}{\delta}\int_0^\infty x^k n_{A,\infty}(x)\,\mathrm{d}x &= k!\int_A^\infty \frac{(t-\beta)^\alpha}{t^{k+1}}\,\mathrm{d}t \\ &\sim k!\int_A^\infty t^{\alpha-k-1}\,\mathrm{d}t = \frac{k!A^{\alpha-k}}{k-\alpha}.\end{aligned}$$

Hence $\kappa_k^* \to 0$ for $k > 2$ and obviously, $\kappa_1^* = 0$, $\kappa_2^* = 1$. Thus all cumulants and hence all moments of $X_{A,\infty}^*(1)$ converge to those of the standard normal r.v. $W(1)$. This implies $X_{A,\infty}^*(1) \xrightarrow{\mathcal{D}} W(1)$ (e.g. [22, Exercise 11 p. 101]), from which the asserted convergence in function space follows from Chap. 15 in [22]. □

Gamma distributions fitted by (Γ_1) or (Γ_2) are appealing alternatives to the normal approximation and perform again significantly better in the numerical examples to be given in Sect. 6. A gamma form of $n_{A,\infty}(x)$ comes up directly: one can use (16) and standard asymptotics of the upper incomplete gamma function to infer that

$$\begin{aligned}n_{A,\infty}(x) &\sim \int_A^\infty \mathrm{e}^{-tx}\delta t^\alpha/\Gamma(1+\alpha)\,\mathrm{d}t = \frac{\delta}{x^{1+\alpha}\Gamma(1+\alpha)}\int_{Ax}^\infty \mathrm{e}^{-y}y^\alpha\,\mathrm{d}y \\ &\sim \frac{\delta}{x^{1+\alpha}\Gamma(1+\alpha)}(Ax)^\alpha\mathrm{e}^{-Ax} \sim \frac{\delta}{\Gamma(1+\alpha)}\frac{A^\alpha}{x}\mathrm{e}^{-Ax}\end{aligned}$$

for any given fixed x. However, the first $\sim$ is not valid if Ax is small or moderate, and in fact the gamma distribution with shape parameter $\delta A^\alpha/\Gamma(1+\alpha)$ and rate parameter A substantially underestimates the order of $X_{A,\infty}(1)$. For example, its mean is 1.2 for $\alpha = 0.8$, $\kappa_2 = 2$, $K = 2$ and $\rho = 0.75$, whereas the correct value is $\mathbb{E}X_{A,\infty}(1) = 5.5$.

6 Numerical Examples

As illustration of the ε- and CM-algorithms, we considered spectrally positive TS processes with varying parameters. Such a process can be parametrized with the variance κ_2, the kurtosis K and α, and one then has

$$\beta = \sqrt{\frac{(2-\alpha)(3-\alpha)}{\kappa_2^2 K}}\,, \quad \delta = \frac{\kappa_2}{\Gamma(2-\alpha)}\beta^{2-\alpha}\,,$$

cf. [3]. We considered three values 0.2, 0.8, 1.4 of α and three 1/2, 2, 8 of K, and normalized by taking $\kappa_2 = 1$. We further considered the normal as well as the two gamma approximations (Γ_1), (Γ_2) of the small jumps, and as measure of performance, we took the L^2–distance

$$d \;=\; \int_0^\infty \big(f(x) - \widehat{f}(x)\big)^2 \mathrm{d}x \tag{18}$$

between the true density $f(x)$ of $X(1)$ and an estimate $\widehat{f}(x)$ provided by simulation. Here $f(x)$ was evaluated by (12), using the MATLAB routines for stable distributions. For $\widehat{f}(x)$, we simulated $M = 10^6$ replicates $Z_1, \ldots, Z_M$ of $X_{\varepsilon,\infty}(1)$ and used the conditional Monte Carlo estimator

$$\widehat{f}(x) \;=\; \frac{1}{M}\sum_{m=1}^{M} \xi(x - Z_m) \tag{19}$$

where $\xi(\cdot)$ is the density in the approximation in question for the density of $X_{(0,\varepsilon)}(1)$. Cf. e.g. [4, p. 146] and [2] (see also [26] for more sophisticated applications of the technique), but note also that conditional Monte Carlo can not universally replace generation of a r.v. distributed according to $\xi(\cdot)$; e.g., this is needed when simulating a discrete skeleton. Numerically, (18) was computed by a discrete approximation with step length 0.01 in the interval $\mathbb{E}X(1) \pm 3$ (recall that $X(1)$ was normalized to standard deviation 1).

The truncation parameters ε, resp. A, for the two algorithms were chosen such that the variance of the approximated small jumps equaled various fractions ρ of the total variance $\kappa_2 = 1$ of all jumps. For the ε-algorithm, this means that for a given ρ

$$\rho \;=\; \int_0^\varepsilon x^2 \frac{\delta \mathrm{e}^{-\beta x}}{x^{1+\alpha}}\,\mathrm{d}x \;=\; \frac{\delta}{\beta^{2-\alpha}}\int_0^{\varepsilon\beta} y^{2-\alpha-1}\mathrm{e}^{-y}\,\mathrm{d}y = \frac{\delta}{\beta^{2-\alpha}}\Gamma(\varepsilon\beta, 2-\alpha)\Gamma(2-\alpha)$$

where $\Gamma(\cdot; 2-\alpha)$ is the lower incomplete Gamma function with parameter $2-\alpha$. Thus

$$\varepsilon \;=\; \frac{1}{\beta}\Gamma^{-1}\left(\frac{\rho\beta^{2-\alpha}}{\delta\Gamma(2-\alpha}; \; 2-\alpha\right)\,.$$

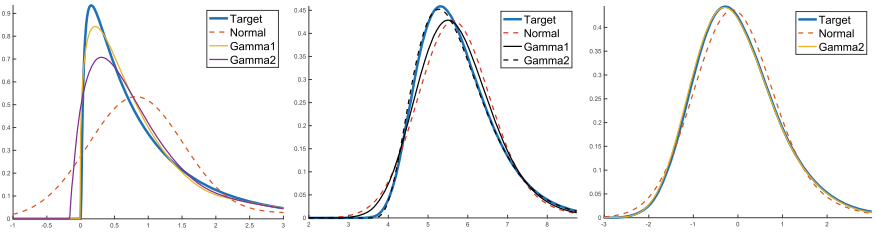


Fig. 2 Left: ε-alg., $\alpha = 0.2$, $K = 2$, $\rho = 0.50$, $d_N = 1.3\,e^{-1}$, $d_{\Gamma 1} = 6.7\,e^{-3}$, $d_{\Gamma 2} = 3.5\,e^{-2}$; middle: CM-alg,. $\alpha = 0.8$, $K = 8$, $\rho = 0.75$, $d_N = 7.6\,e^{-3}$, $d_{\Gamma 1} = 2.6\,e^{-3}$, $d_{\Gamma 2} = 5.3\,e^{-5}$ right: ε-alg., $\alpha = 1.4$, $K = 2$, $\rho = 0.75$, $d_N = 2.0\,e^{-2}$, $d_{\Gamma 2} = 1.1\,e^{-4}$

For the CM-algorithm, we have instead by (17) that

$$\rho = \int_0^\infty x^2\,\mathrm{d}x \int_A^\infty \mathrm{e}^{-tx}\frac{\delta(t-\beta)^\alpha}{\Gamma(1+\alpha)}\,\mathrm{d}t = \frac{\delta}{\Gamma(1+\alpha)}\int_A^\infty \frac{(t-\beta)^\alpha}{t^3}\,\mathrm{d}t$$
$$= \frac{\delta}{\Gamma(1+\alpha)}\int_B^\infty \frac{y^\alpha}{(\beta+y)^3}\,\mathrm{d}y$$

where $B = A - \beta$, and this equation was solved numerically.

Here S is the skewness of $X(1)$ and λ is the Poisson mean in the compound Poisson sum of the simulated "large" jumps, that is,

$$\lambda = \int_\varepsilon^\infty n(x)\,\mathrm{d}x = \int_\varepsilon^\infty \frac{\delta \mathrm{e}^{-\beta x}}{x^{1+\alpha}}\,\mathrm{d}x\,, \quad \lambda = \int_0^A \frac{v(t)}{t}\,\mathrm{d}t = \int_0^B \frac{\delta t^\alpha}{(t+\beta)\Gamma(1+\alpha)}\,\mathrm{d}t$$

in the two cases. The L_2 distances in (18) are denoted by d_N for the normal approximation and by d_{Γ_1}, d_{Γ_2} for the two gamma ones. Graphs of $f(x)$ and the $\widehat{f}(x)$ are in Fig. 2 for some selected the parameter combinations in Table 2.

Our interpretation of Fig. 2 is that an L_2-distance of e^{-4} or less corresponds to an almost perfect fit, whereas one of order e^{-3} is sufficient for most practical purposes, one of order e^{-2} or more inadequate. With this in mind, we were quite surprised to see how well both algorithms perform already for so large values of ρ as 75% and 50%, or equivalently for so small values of λ as those reported in the Tables 1 and 2. One further notes that both algorithms improve as K gets smaller or α larger, which is in agreement with limit theorems given in [3] stating roughly that the distribution of $X(1)$ gets closer to normal in the two cases.

Taking λ as measure of computational effort is certainly not unambiguous. On top comes the effort in generating from the r.v.'s $Y_n(\varepsilon)$, $Y_n(A)$ with densities proportional to $n(x)$, $\varepsilon < x < \infty$, resp. $v(t)/t$, $0 < t < A$. However, this issue is largely implementation dependent. We have given one suggestion (based on (15)) for the ε-algorithm in Sect. 4 and give a similar A-R scheme for the CM-algorithm and TS case in the appendix. Both are certainly amenable to improvement. Comparison of the ε- and CM algorithms show that λ is slightly higher for the CM algorithm. However, the values of λ reported in the tables are quite small and thus $1 + \lambda$ could

Table 1 $\rho = 75\%$

α	K	S	ε-algorithm				CM-algorithm			
			λ	d_N	d_{Γ_1}	d_{Γ_2}	λ	$d(F, F_N)$	$d(F, F_{\Gamma_1})$	$d(F, F_{\Gamma_2})$
0.2	1/2	0.57	0.23	$1.4\,e^{-3}$	$8.2\,e^{-5}$	$4.0\,e^{-5}$	1.6	$2.0\,e^{-2}$	$2.8\,e^{-5}$	$4.0\,e^{-5}$
	2	1.13	0.06	$1.4\,e^{-2}$	$1.7\,e^{-4}$	$2.2\,e^{-4}$	0.40	$1.5\,e^{-2}$	$2.8\,e^{-4}$	$1.4\,e^{-4}$
	8	2.27	0.01	$1.6\,e^{-1}$	$1.3\,e^{-2}$	$3.7\ e^{-2}$	0.10	$1.6\,e^{-1}$	$2.6\,e^{-2}$	$7.1\,e^{-2}$
0.8	1/2	0.52	0.24	$8.8\,e^{-4}$	$2.3\,e^{-4}$	$4.3\,e^{-5}$	1.3	$1.2\,e^{-2}$	$4.6\,e^{-4}$	$5.3\,e^{-5}$
	2	1.04	0.06	$7.6\,e^{-3}$	$2.6\,e^{-3}$	$5.3\,e^{-5}$	0.33	$8.2\,e^{-3}$	$3.5\,e^{-3}$	$3.3\,e^{-4}$
	8	2.09	0.01	$5.0\,e^{-2}$	$2.4\,e^{-2}$	$2.8\,e^{-3}$	0.08	$5.0\,e^{-2}$	$2.5\,e^{-2}$	$2.0\,e^{-2}$
1.4	1/2	0.43	0.30	$2.9\,e^{-4}$	–	$2.2\,e^{-5}$	1.23	$3.7\,e^{-4}$	–	$4.8\,e^{-5}$
	2	0.77	0.07	$2.0\,e^{-3}$	–	$1.1\,e^{-4}$	0.31	$2.4\,e^{-3}$	–	$2.2\,e^{-4}$
	8	1.73	0.02	$1.1\,e^{-2}$	–	$1.0\,e^{-3}$	0.08	$1.1\,e^{-2}$	–	$3.6\,e^{-3}$

Table 2 $\rho = 50\%$

α	K	S	ε-algorithm				CM-algorithm			
			λ	d_N	d_{Γ_1}	d_{Γ_2}	λ	$d(F, F_N)$	$d(F, F_{\Gamma_1})$	$d(F, F_{\Gamma_2})$
0.2	1/2	0.57	0.96	$2.9\,e^{-4}$	$8.8\,e^{-5}$	$3.9\,e^{-5}$	4.1	$5.7\,e^{-4}$	$2.8\,e^{-5}$	$4.6\,e^{-5}$
	2	1.13	0.24	$6.1\,e^{-3}$	$4.4\,e^{-4}$	$1.4\,e^{-4}$	1.0	$7.5\,e^{-3}$	$1.4\,e^{-4}$	$1.0\,e^{-4}$
	8	2.27	0.060	$1.3\,e^{-1}$	$6.7\,e^{-3}$	$3.5\,e^{-2}$	0.26	$1.3\,e^{-1}$	$1.9\,e^{-2}$	$4.4\,e^{-2}$
0.8	1/2	0.52	1.20	$1.2\,e^{-4}$	$2.6\,e^{-5}$	$4.3\,e^{-5}$	4.35	$2.5\,e^{-4}$	$8.0\,e^{-5}$	$3.6\,e^{-5}$
	2	1.04	0.30	$5.1\,e^{-4}$	$5.1\,e^{-4}$	$4.7\,e^{-5}$	1.09	$2.9\,e^{-3}$	$1.1\,e^{-3}$	$1.2\,e^{-4}$
	8	2.09	$7.5\,e^{-2}$	$2.3\,e^{-2}$	$1.1\,e^{-2}$	$6.2\,e^{-4}$	$2.7\,e^{-2}$	$3.0\,e^{-2}$	$1.4\,e^{-2}$	$4.2\,e^{-3}$
1.4	1/2	0.43	2.42	$2.0\,e^{-5}$	–	$4.0\,e^{-5}$	7.66	$2.8\,e^{-5}$	–	$3.0\,e^{-5}$
	2	0.77	0.61	$1.8\,e^{-4}$	–	$6.1\,e^{-5}$	1.92	$3.1\,e^{-4}$	–	$5.8\,e^{-5}$
	8	1.73	0.15	$2.3\,e^{-2}$	–	$1.3\,e^{-4}$	0.48	$2.9\,e^{-3}$	–	$3.1\,e^{-4}$

be a more fair measure than λ, taking into account also the generation of the Poisson r.v.'s in addition to the Y. This makes the difference even smaller. As for precision, values of order $e-5$ should not be compared as they do not improve by increasing ρ, which could presumably be due to the discretization. Once this is said, the Γ_2 scheme gives most often better precision than the Γ_1 one, and both improve upon the normal, in some cases even significantly. The ε-algorithm gives slightly more precise estimates for the given ρ than the CM one, but most often not that much. Altogether, which one to prefer may depend on case-dependent issues such as the facility to generate the $Y_n(\varepsilon)$ or $Y_n(A)$.

Concerning the chosen values $1/2, 2, 8$ of the kurtosis K, we remark that in financial log-return data K is most often of order 1–3 for daily log-returns series, but higher values occur when calibrating parameters, cf. Table 1 in [3]. Sampling at higher frequencies than daily will also increase K, and hence one may expect that larger values of λ than the ones in our tables will be needed for good precision.

7 Exact Simulation of $X(h)$ and other Methods

In our main examples, it is fairly straightforward to generate a r.v. distributed as $X(h)$ in a NIG process. Indeed, one description of the process is as subordinate to a BM W with drift β w.r.t. an inverse Gaussian subordinator $\chi(t)$. In more detail, if W_1 is another independent BM with drift γ and $\chi(t) = \inf\{s > 0 : W_1(t) > \delta t\}$, then $W(\chi(t)) + \mu t$ is distributed as $X(1)$ in a NIG$(\delta, \alpha, \beta, \mu)$ process with $\alpha = \sqrt{\beta^2 + \gamma^2}$. Here a r.v. distributed as $\chi(t)$ need not be simulated via the relation to W_1 but can be directly generated. For $X(h)$, just replace δ by δh and μ my μh. These facts are surveyed in, e.g., [4, p. 343] and implemented in, e.g., [25]. A similar but easier exact subordination construction applies to the VG process. Asymptotic subordination algorithms for TS and MX are in [27].

For the spectrally positive TS process with finite variation ($\alpha < 1$), it was noted in [3] that a r.v. distributed as $X(1)$ can be generated by an A-R scheme, using (12) with the $S_\alpha(\sigma, 1, 0)$ r.v. Z as proposal and acceptance probability $\mathrm{e}^{-\beta z}$ when $Z = z$; for the standard algorithm to generate Z, see [4, p. 332]. Two-sided processes are of course generated by taking the difference between the positive and negative parts. The simplicity of this scheme should be compared to other approaches in the literature, e.g. [8, 23]. It was also remarked in [3] that the situation is more complicated when $\alpha \geq 1$, since then $X(1)$ is supported by the whole of $\mathbb{R}$ and $\mathrm{e}^{-\beta z}$ is unbounded there. We suggest here an exact scheme based on asymptotic properties of stable densities. The details are in the Appendix but are included more for the sake of completeness than because we think the scheme is more attractive than the simple and efficient ε- and CM-algorithms.

A general comment on the method of discrete skeletons is that it gives little information on the whole path $X(0:T)$ unless one uses a skeleton with a quite small h and thereby a considerable computational effort.

We are not aware of methods for exact simulation of $X(h)$ in the MX process. One could potentially use the explicit form of the density, cf. (9), via A-R, but a difficulty is to find suitable bounds for the complex gamma function.

Another approximate method is based on using a series expansion of the form $X(T) = \sum_1^\infty \{H(\Gamma_n, V_n) - c_n T\}$ where the Γ_n are the order epochs of a standard Poisson process, and the V_n independent i.i.d. (possibly multivariate) r.v.'s., see the surveys in [31] and [4] XII.4. In the implementation, ones truncates to $n \leq N$ terms. Since $H(\cdot, v)$ is typically decreasing for fixed v, this method is hardly intrinsically different from the ε-algorithm. Calculation of H is not always straightforward. We are not aware of systematic studies of the error term $\sum_{N+1}^\infty \ldots$.

8 Maxima, Minima and Other Path Functionals

In Sects. 4–6 and 7, we have concentrated on simulation of $X(T)$ alone, say $T = h$ or $T = 1$ (there is no loss of generality in taking $T = 1$ since $X(T) = X_T(1)$ where X_T is the process obtained by replacing the Lévy measure ν by $T\nu$). In the financial

context, this covers European options, where Ψ in (1) is a function of $X(T)$ alone. E.g. $\Psi\big(X(0:T) = \mathrm{e}^{-rT}\big[Z(0)\mathrm{e}^{X(T)} - K\big]^+$ for a European call with strike K. For many other options, Ψ is, however, more complicated. E.g. for an down-and-in barrier option

$$\Psi\big(X(0:T)\big) \;=\; \mathrm{e}^{-rT}\big[Z(0)\mathrm{e}^{X(T)} - K\big]^+ \mathbb{I}\big(Z(0)\mathrm{e}^{X(t)} \le L \text{ for some } t \le T\big)\,.$$

One therefore needs to know also the minimum $m_T = \inf_{t\le T} X(t)$ of $X(0:T)$, which typically is close to the value at some negative jump. Minima or maxima also come up in the context of queues modeled by Lévy input, where key processes Y such as workload, queue length etc. are obtained by reflecting the input X at 0. This means

$$Y(T) \;=\; \big(Y(0) + X(T)\big) \vee \max_{t\le T}\big(X(T) - X(t)\big)\,.$$

In particular, $Y(T) \stackrel{d}{=} M_T$ where $M_T = \sup_{t\le T} X(t)$ in the case $Y(0) = 0$ of an initially empty queue. If X is simulated as a discrete skeleton with step size h, the path of Y is approximated by $Y_h(0) = Y(0)$ and the Lindley recursion

$$Y_h\big((n+1)h\big) \;=\; \big[Y_h(nh) + X\big((n+1)h\big) - X(nh)\big]^+\,,$$

leading to

$$Y_h(Nh) \;=\; \big(Y(0) + X(Nh)\big) \vee \max_{n\le N}\big(X(Nh) - X(nh)\big) \;\stackrel{d}{=}\; \max_{n\le N} X(nh)$$

where the final $\stackrel{d}{=}$ requires $Y(0) = 0$. For these facts, see Sects. III.6–7, IX.2 of [1].

We mention several strategies to access a minimum or maximum, say $m(T)$, without recommending any particular one (in fact, such a choice may depend on the particular application context and a more extensive numerical study). One strategy is just to simulate a sufficiently fine skeleton exactly, when possible, and then take the minimum along the skeleton. This may be supplemented with continuity corrections as developed in [5, 20], that is, r.v.'s approximating

$$\min_{nh\le t\le (n+1)h} X(t) \;\big|\; X_{(n+1)h}, X_{nh}\,.$$

If exact simulation of a skeleton is not feasible, one may instead generate the skeleton approximately by one of the compound Poisson algorithms of Sects. 4, 5, allocate uniform $[0, T]$ locations to the Poisson jump times τ_n , and supplement the minimum along the τ_n by invoking bridge r.v.'s of the form

$$\min_{\tau_n\le t\le \tau_{n+1}} \big(\widehat{X}_{0:\varepsilon}(t) - \widehat{X}_{0:\varepsilon}(\tau_n)\big) \;\big|\; \widehat{X}_{0:\varepsilon}(\tau_n)\,.$$

The distribution and hence generation of such bridge minima is standard when $\widehat{X}_{0:\varepsilon}$ is generated by using the normal approximation. For our gamma approximations, they may be efficiently generated by invoking the relation between gamma, beta and Dirichlet distributions, as developed in [7] and implemented in [25].

Similar remarks apply to other and possibly more complicated path functionals. For example, for a Parisian option one needs to know the first time the asset price $e^{X(t)}$ makes an excursion of length $> D$ below some level L.

Acknowledgements I am grateful to the reviewers for useful comments and corrections, and to Alexey Kuznetsov for useful hints related to Example 1.

Appendix

An A-R Scheme for Generation from $v(t)/t$ in the TS Case

We need to generate a r.v. Z with density proportional to $u(t)/t$, $\beta < t < A$ where $u(t) = (t-\beta)^\alpha$. To this end, write $Z = \beta + Z_0$ where Z_0 has density proportional to $u(t+\beta)/(t+\beta) = t^\alpha/(t+\beta)$, $0 < t < B$ where $B = A - \beta$. Here $Y = 1/Z_0$ has density proportional to $1/y^{1+\alpha}/(1+\beta y)$, $1 < 1/B < y < \infty$, and can therefore be generated by A-R with either a Pareto(α) proposal and acceptance probability proportional to $1/(1+\beta y)$ (high for small y) or a Pareto($1+\alpha$) proposal and acceptance probability proportional to $y/(1+\beta y)$ (high for large y). As in Sect. 4, we use a mixture, corresponding to breaking the compound Poisson part in 2) above into two. So, let

$$\lambda_1 = \int_{1/B}^{y_0} \frac{1}{y^{1+\alpha}(1+\beta y)}\,\mathrm{d}y\,, \quad \mu_1 = \int_{1/B}^{y_0} \frac{1}{y^{1+\alpha}}\,\mathrm{d}y = \frac{1}{\alpha}[B^\alpha - 1/y_0^\alpha]\,,$$
$$\lambda_2 = \int_{y_0}^{\infty} \frac{1}{y^{1+\alpha}(1+\beta y)}\,\mathrm{d}y\,, \quad \mu_2 = \int_{y_0}^{\infty} \frac{1}{y^{2+\alpha}}\,\mathrm{d}y = \frac{1}{(1+\alpha)y_0^{1+\alpha}}\,,$$
$$C_1 = \frac{\mu_1}{\lambda_1(1+\beta/B)}\,, \quad C_2 = \frac{\mu_2}{\beta\lambda_2}$$

The target densities are then

$$f_1(y) = \frac{1}{\lambda_1 y^{1+\alpha}(1+\beta y)},\ 1/B < y < y_0, \text{ and } f_2(y) = \frac{1}{\lambda_2 y^{1+\alpha}(1+\beta y)},\ y_0 < y < \infty,$$

and chosen with probabilities $\lambda_1/(\lambda_1+\lambda_2)$, resp. $\lambda_2/(\lambda_1+\lambda_2)$, and the proposals are

$$g_1(y) = \frac{1}{\mu_1 y^{1+\alpha}},\ 1/B < y < y_0, \text{ and } g_2(y) = \frac{1}{\mu_2 y^{2+\alpha}},\ y_0 < y < \infty,$$

Then $f_1(y) \le C_1(y_0)g_1(y)$ and $f_2(y) \le C_2(y_0)g_2(y)$, and we may use A-R with acceptance probabilities

$$\frac{f_1(y)}{C_1(y_0)g_1(y)} = \frac{1}{1+\beta y}, \quad \frac{f_2(y)}{C_2(y_0)g_2(y)} = \frac{y}{1+\beta y}$$

for r.v. generation from f_1, resp. f_2. This gives expected numbers $C_1(y_0)$, $C_2(y_0)$ of samplings from $g_1(y)$, resp. $g_2(y)$, and as measure $E(y_0)$ of the computational effort, we shall use the total number of these samplings, i.e.

$$E(y_0) \;=\; \lambda_1 C_1 + \lambda_2 C_2 \;=\; \frac{\mu_1}{1+\beta/B} + \frac{\mu_2}{\beta} \;=\; \frac{\beta\mu_1 + (1+\beta/B)\mu_2}{\beta(1+\beta/B)} \qquad (20)$$

Proposition 4 *The function $E(y_0)$, $1/B < y_0 < \infty$, is minimized for $y_0 = y_0^* = 1/\beta + 1/B$.*

Proof In (20), β and B as well as term B^α/α in μ_1 do not depend on y_0, so we are left with the minimization of $-\beta/\alpha/y_0^\alpha + (1+\beta/B)/(1+\alpha)/y_0^{1+\alpha}$. The derivative is $1/y_0^{1+\alpha}/(1+\beta/B) - 1/y_0^{2+\alpha}$ which changes sign from negative to positive at y_0^*. From this the result follows. □

An A-R Scheme for Spectrally Positive Infinite Variation TS Processes

Let f_0 be the density of a strictly α-stable distribution $S_\alpha(\sigma, 1, 0)$ distribution with $\sigma = \big(-\delta\Gamma(-\alpha)\cos(\pi\alpha/2)\big)^{1/\alpha}$. The goal is to generate a r.v. X from the density $f(x) = \exp\{-\beta x - \psi\} f_0(x)$ in the case $\alpha > 1$ where f and f_0 have support on the whole of $\mathbb{R}$; here $\psi = \delta\Gamma(-\alpha)\beta^\alpha$. We use that $f_0(x)$ has the asymptotics [29, p. 100]

$$f_0(x) \;\sim\; \frac{c_1}{|x|^\ell} \exp\big\{-c_2|x|^\eta\big\} \;\text{ as } x \to -\infty \qquad (21)$$

for suitable (explicit) constants) c_1, c_2 and $\ell = \alpha/(2\alpha-2)$, $\eta = \alpha/(\alpha-1)$.

For initialization of the algorithm:

(1) Select $-A < 0$ and compute $p = \int_{-A}^{\infty} f(x)\,\mathrm{d}x$

(2) Select $c_3 < c_2$ and find $c_4 < \infty$ such that

$$h(x) \;=\; \frac{\mathrm{e}^{\beta|x|} f_0(x)}{|x|^{\eta-1}\exp\big\{-c_3|x|^\eta\big\}} \;\le\; c_4 \quad \text{for all } x < -A.$$

The algorithm is then as follows:

(3) Generate I as Bernoulli(p).

(4) If $I = 1$, generate $X \in (-A, \infty)$ having density $f(x)/p$, $-A < x < \infty$, by A-R with proposal Z_0 having a strictly α-stable distribution $S_\alpha(\sigma, 1, 0)$ conditioned to $(-A, \infty)$ and acceptance probability $\mathrm{e}^{-b(z+A)}$ when $Z_0 = z$
(5) If $I = 0$, generate $X \in (-\infty, -A)$ with density $\widetilde{f}(x) = \mathrm{e}^{b|x|-\psi} f_0(x)/(1-p)$, $= \infty < x < -A$, by an A-R scheme defined as follows. As proposal, take a r.v. Z_1 distributed as $-Z_2$ given $Z_2 > A$ where $Z_2 > 0$ is Weibull with $\mathbb{P}(Z_2 > z) = \mathrm{e}^{-c_3 z^\eta}$. If $Z_1 = x$, accept w.p. $c_4 h(x)$.
(8) return X.
Explanation: Step (2) is possibly because (21), $c_2 > c_3$ and $\eta > 1$ imply $h(x) \to 0$ as $x \to -\infty$. In (5), the proposal density is $g(x) = \mathbb{P}\big(X_2 \in \mathrm{d}|x| \,\big|\, Z_2 > A\big) = c_3\eta|x|^{\eta-1}\mathrm{e}^{-c_3 z^\eta}/\mathrm{e}^{-c_3 A^\eta}$. Thus the ratio of the target density to the proposal density is

$$\frac{\widetilde{f}(x)}{g(x)} \;=\; c_5 h(x) \quad \text{where} \quad c_5 = \frac{\exp\{-\psi - c_3 A^\eta\}}{c_3\eta(1-p)}$$

Hence $\widetilde{f}(x)/g(x) \le c_0 h(x)$ where $c_0 = c_4 c_5$, and acceptance w.p. $\widetilde{f}(x)/c_0/g(x) = c_4 h(x)$ will produce the correct result. The conditioned sampling of proposals in (6) and (7) is straightforward by sampled by sampling a $S_\alpha(\sigma, 1, 0)$, resp. Weibull, r.v. until the conditioning requirement is met. Available software, say MATLAB or Nolan's STABLE package (see the Preface to [29]) accounts for computing $f_0(x)$ and generating the $S_\alpha(\sigma, 1, 0)$ r.v.'s. The Weibulls can be generated by inversion.

References

1. Asmussen, S.: Applied Probability and Queues, 2nd edn. Springer, New York (2003)
2. Asmussen, S.: Conditional Monte Carlo for sums, with applications to insurance and finance. Ann Actuarial Sci **12**, 455–478 (2018)
3. Asmussen, S.: On the role of skewness and kurtosis in tempered stable (CGMY) Lévy models in finance. Finance and Stochastics **26**, 383–416 (2022)
4. Asmussen, S., Glynn, P.: Stochastic Simulation, Algorithms and Analysis. Springer, Berlin (2007)
5. Asmussen, S., Ivanovs, J.: Discretization error for a two-sided reflected Lévy process. Queueing Syst. **89**, 199–212 (2018)
6. Asmussen, S., Rosiński, J.: Approximations for small jumps of Lévy processes with a view towards simulation. J. Appl. Probab. **38**, 482–493 (2001)
7. Avramidis, A., L'Ecuyer, P., Tremblay, P.A.: Efficient simulation of Gamma and variance Gamma processes. In: Proceedings of the 2003 Winter Simulation Conference, pp. 319–326 (2003)
8. Ballotta, L., Kyriakou, I.: Monte Carlo simulation of the CGMY process and option pricing. J. Futur. Mark. **4**, 1095–1121 (2014)
9. Barndorff-Nielsen, O.: Processes of normal inverse Gaussian type. Financ. Stochast. **4**, 1095–1121 (1998)
10. Bertoin, J.: Lévy Processes. Cambridge University Press (1996)
11. Bondesson, L.: On simulation from infinitely divisible distributions. Adv. Appl. Probab. **14**, 855–869 (1982)
12. Carr, P., Geman, H., Madan, D., Yor, M.: The fine structure of asset returns: an empirical investigation. J. Bus. **75**, 305–332 (2002)

13. Carr, P., Madan, D.: Option valuation using the fast Fourier transform. J. Comput. Financ. **2**, 61–73 (1998)
14. Cont, R., Tankov, P.: Financial Modelling with Jump Processes. Chapman and Hall/CRC (2004)
15. Debicki, K., Mandjes, M.: Queues and Lévy Fluctuation Theory. Springer (2018)
16. Dia, E.: Error bounds for small jumps of Lévy processes. Adv. Appl. Probab. **45**, 86–105 (2013)
17. Feller, W.: An Introduction to Probability Theory and Its Applications, vol. 2, 2 edn. Wiley (1971)
18. Grigelionis, B.: Generalized z-distributions and related stochastic processes. Lietuvos Matematikos Rinkinys **41**, 239–251 (2001)
19. Hackmann, D., Kuznetsov, A.: Approximating Lévy processes with completely monotone jumps. Ann. Appl. Probab. **26**, 328–359 (2016)
20. Ivanovs, J.: Zooming in on a Lévy process at its supremum. Ann. Appl. Probab. **28**, 912–940 (2018)
21. Jeannin, M., Pistorius, M.: A transform approach to compute prices and Greeks of barrier options driven by a class of Lévy processes. Quant. Financ. **10**, 629–644 (2010)
22. Kallenberg, O.: Foundations of Modern Probability, 2nd ed. edn. Springer (2003)
23. Karlsson, P.: Finite element based Monte Carlo simulation of options on Lévy driven assets. Int. J. Financ. Eng. **5**, 1850013 (2018)
24. Küchler, U., Tappe, S.: Tempered stable distributions and processes. Stoch. Process. Their Appl. **123**, 4256–4293 (2013)
25. L'Ecuyer, P.: `https://github.com/umontreal-simul/ssj` (2015). Software package, Université de Montreal
26. L'Ecuyer, P., Puchhammer, F., Ben Abdellah, A.: Monte Carlo and quasi-Monte Carlo density estimation via conditioning. Inf. J. Comput. (2022). https://doi.org/10.1287/ijoc.2021.1135
27. Madan, D., Yor, M.: "CGMY and Meixner subordinators are absolutely continuous with respect to one sided stable subordinators". HAL Archives-Ouvert.fr hal-00016662v2 (2006)
28. Mozzola, M., Muliere, P.: Reviewing alternative characterizations of Meixner process. Probab. Surv. **8**, 127–154 (2011)
29. Nolan, J.: Univariate Stable Distributions Models for Heavy Tailed Data. Springer, Berlin (2020)
30. Poirot, J., Tankov, P.: Monte Carlo option pricing for tempered stable (CGMY) processes. Asia-Pac. Financ. Mark. **13**, 327–344 (2006)
31. Rosiński, J.: Series representations of Lévy processes from the perspective of point processes. In: Barndorff-Nielsen, O., Mikosch, T., Resnick, S. (eds.) Lévy Processes — Theory and Applications, pp. 401–415. Birkhäuser (2001)
32. Rydberg, T.: The normal inverse Gaussian Lévy process: simulation and approximations. Stoch. Model. **13**, 887–910 (1997)
33. Sato, K.: Lévy Processes and Infinitely Divisible Distributions. Cambridge University Press (1999)
34. Schilling, R., Song, R., Vondraček, Z.: Bernstein Functions. Theory and Applications, 2nd edn. de Gruyter (2012)
35. Schoutens, W.: Lévy Processes in Finance Pricing Financial Derivatives. Wiley (2003)
36. Seneta, E.: A Tauberian theorem of E. Landau and W. Feller. Annals of Probability **6**, 1057–1058 (1973)
37. Signahl, M.: On error rates to normal approximations and simulation schemes for Lévy processes. Stoch. Model. **19**, 287–298 (2003)

Exact Sampling for the Maximum of Infinite Memory Gaussian Processes

Jose Blanchet, Lin Chen, and Jing Dong

Abstract We develop an exact sampling algorithm for the all-time maximum of Gaussian processes with negative drift and general covariance structures. In particular, our algorithm can handle non-Markovian processes even with long-range dependence. Our development combines a milestone-event construction with rare-event simulation techniques. This allows us to find a random time beyond which the running time maximum will never be reached again. The complexity of the algorithm is random but has finite moments of all orders. We also test the performance of the algorithm numerically.

Keywords Perfect simulation · Infinite memory · Rare event sampling

1 Introduction

It is a pleasure to contribute this paper in honor of Professor Pierre L'Ecuyer, whose many contributions to stochastic simulation have significantly influenced this area. In this paper, we propose and study an exact simulation algorithm for the infinite-horizon maximum of some general Gaussian processes. What makes our contribution novel relative to other results in the literature is the non-Markovian, even long-range dependence structure of the processes we study. *Long-range dependence* is an important phenomenon that arises in many applications. An early work studying such a phenomenon dates back to the 1950's when Hurst [29] studied the flow of water in Nile river. Since then, evidence of long-range dependence has been found in economics [36], internet traffic [8, 31], linguistics [3], etc. (see [23, 37]

J. Blanchet (✉)
Stanford University, 475 Via Ortega, Stanford, CA 94305, USA
e-mail: jose.blanchet@stanford.edu

L. Chen · J. Dong
Columbia University, 3022 Broadway, New York city, NY 10027, USA
e-mail: lc3110@columbia.edu

J. Dong
e-mail: jing.dong@gsb.columbia.edu

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_3

for comprehensive reviews). More generally, Gaussian processes have been one of the main modeling tools to capture various non-Markovian features [7, 35, 37]. For example, in contrast to long-range dependence, they are also used to model a phenomenon associated with very rough paths that have rapid short-term fluctuations. This phenomenon is found in volatility of high frequency data [6, 25]. Due to the complicated dependences, limited analytical results are available for sample-path quantities (such as the maximum) of these processes. Simulation has been a key numerical tool for analysis. The all-time maximum of Gaussian processes arises in communication applications [1], risk management, and analysis of queues [5].

Let $\mathcal{S} = \{S_n : n \in \mathbb{Z}^+\}$ be a discrete-time Gaussian process. We assume $S_0 \equiv 0$, $\mathbb{E}[S_n] = -n\mu$ for some $\mu > 0$, i.e., the process has a negative drift, and

$$\text{Var}(S_n) = \sigma^2 n^{2H} + o(n^{2H}) \text{ for } H \in (0, 1) \text{ and } \sigma^2 > 0. \tag{1}$$

Note that $\text{Var}(S_n)$ can grow sublinearly ($H < 1/2$) or superlinearly ($H > 1/2$) in n.

The Gaussian process $\mathcal{S}$ defined above can be fairly general. Let $X_n = S_n - S_{n-1}$ denote the increment of the Gaussian process. In the special case where $\{X_n : n \in \mathbb{Z}^+\}$ is a stationary process with mean $-\mu$ and variance σ^2, we have

$$\text{Var}(S_n) = \sum_{i=1}^{n}\sum_{j=1}^{n} \text{Cov}(X_i, X_j) = \sigma^2 \left(n + 2\sum_{i=1}^{n-1}(n-i)\rho_i \right),$$

where $\rho_i := \text{Corr}(X_n, X_{n+i})$ for $n \in \mathbb{Z}^+$. When $\rho_i = i^{-\alpha}$ for $\alpha \in (0, 1)$, $\text{Var}(S_n) = Cn^{2-\alpha} + o(n^{2-\alpha})$ for some $C > 0$. In this case, $H > 1/2$ and $\mathcal{S}$ has long-range dependence. In general, a stationary Gaussian process is said to have long-range dependence or infinite memory if $\lim_{n\to\infty} \text{Var}(S_n)/n = \infty$ [27].

A classic stationary Gaussian process that can be used to model both long-range dependence or rapid short-term fluctuations is fractional Brownian motion (fBM). Let $\boldsymbol{B}^H = \{B^H(t) : t \in \mathbb{R}^+\}$ be an fBM with Hurst index $H \in (0, 1)$. $\boldsymbol{B}^H$ is a self-similar Gaussian process with $\mathbb{E}[B^H(t)] = 0$ and covariance function

$$\mathbb{E}[B^H(s)B^H(t)] = \frac{1}{2}\left(t^{2H} + s^{2H} - |t-s|^{2H}\right).$$

When $H > 1/2$, $\boldsymbol{B}^H$ is a non-Markovian Gaussian process with long-range dependence; when $H = 1/2$, $\boldsymbol{B}^H$ is a standard Brownian motion; when $H < 1/2$, $\boldsymbol{B}^H$ is a non-Markovian Gaussian process with very rough paths (even more rough than Brownian motion). We define an embedded discrete-time process $S_n = -n\mu + \sigma B^H(n)$, which is often referred to as fractional Gaussian noise. Then, $\mathbb{E}[S_n] = -n\mu$ and $\text{Var}(S_n) = \sigma^2 n^{2H}$, which satisfies (1).

In this paper, we are interested in estimating the all-time maximum of the Gaussian process $\mathcal{S}$. The all-time maximum is defined as

$$M_\infty = \max_{n\geq 0} S_n.$$

We develop an algorithm to draw samples from the exact distribution of M_∞. The termination time of the algorithm is random but has finite moments of all orders.

Sampling the all-time maximum of a stochastic process is in general a challenging task. Based on the definition of M_∞, naive simulation would require one to generate an infinite sequence $\{S_n : n \geq 0\}$, which is infeasible. Our algorithm combines the construction of a sequence of stopping times with rare-event simulation techniques. It allows us to identify a non-stopping time T, which is defined as as the time beyond which the running-time maximum $M_T = \max_{0 \leq n \leq T} S_n$ will never be reached again, i.e., $S_n < M_T$ for $n > T$. Then, $M_\infty = M_T$.

Our development builds on three key components (1) a milestone event construction, which is used to define the sequence of stopping times; (2) a rare-event simulation technique for Gaussian processes called Target Bridge Sampling (TBS); and (3) a sandwiching construction to sample Bernoulli random variables whose probability of success cannot be evaluated exactly. These techniques are not new and have found important applications in the simulation literature. However, how to apply them jointly to solve our simulation problem is highly non-trivial.

The idea of milestone events is to construct a sequence of stopping times (corresponding to event times) in a way that enables us to translate the infinite-horizon simulation problem to finding the last finite stopping time in the sequence (see Sect. 2 for more details). The strategy was first developed in [13] and has seen applications in many subsequent works [9, 10]. The key to successful implementation of this idea is to construct a proper sequence of milestone events, which requires understanding the large deviation behavior of the underlying stochastic process. Here, we are able to apply this idea by analyzing the tail behavior of the Gaussian processes.

The milestone events are defined in such a way that it becomes increasingly harder to reach them along the sequence, i.e., the probability that the stopping time is finite is decreasing. To efficiently sample the trajectories leading to the milestone events, we utilize a change of measure induced by TBS. TBS was developed in [12]. It is an importance sampling algorithm not based on exponential tilting, which allows us to circumvent the challenge of tracking the most likely path under non-Markovian structures. In [12], it is applied to estimate $\mathbb{P}(M_\infty > b)$ for large values of b, which is different from our task here: drawing exact samples from M_∞. In addition, the implementation in [12] involves a truncation which induces a bias in the estimator. We circumvent the truncation using a sandwiching construction.

The sandwiching construction is also known as the series method [17]. In our application of the method, we want to generate a Bernoulli random variable whose probability of success p cannot be evaluated exactly. The idea is to approximate p from above and below by sequences of refined bounds. Then, we can sequentially update the bounds to determine whether U is less than p, where U is a Uniform random variable (see Sect. 3 for more details). When applying the method, the key is to come up with a carefully-designed sequence of bounds. As we will explain in Sect. 3, due to the general covariance structure, constructing the sequence of upper bounds can be quite involved.

Literature Review. There is a growing amount of literature on generating discrete-time Gaussian processes with complex time correlation. The papers [4, 21] develop efficient exact simulation algorithms for Gaussian process with long-range dependence. They come up with techniques to reduce the computational complexity associated with the high-dimensional covariance matrices. These techniques can be applied in our algorithm as well when generating the underlying Gaussian processes. Efficient simulation of fBM and related processes has also attracted continuous attention from the literature [16, 19, 30, 32, 34] (see also [18] for a review).

As explain above, all-time maximum is in general difficult to simulate exactly. For random walks with independent increments, the paper [24] is among the first to use rare-event simulation techniques to draw exact samples of the maximum when the increments are light-tailed distributed. Later extensions include heavy-tailed increments [14] and the maximum over a nonlinear boundary [11]. For Gaussian processes with general covariance structures, rare-event simulation techniques have been applied to estimate the tail probability of M_∞, i.e., $\mathbb{P}(M_\infty > b)$ for large values of b [2, 12, 15, 20, 28]. Comparing to previous works, substantial new developments are required to draw exact samples from M_∞. Meanwhile, the existing rare-event simulation techniques can be applied as an important intermediate step in our development. The papers [22, 33] develop exact simulation algorithms for max-stable fields at a finite collection of locations, which requires generating the maximum of a Gaussian random field.

Notation. Throughout the paper, we use $\Phi(x)$ to denote the cumulative distribution function (cdf) of standard Gaussian distribution, and denote $\bar{\Phi}(x) := 1 - \Phi(x)$ as the tail cdf. Let $\mathcal{S}_n = (S_1, \ldots, S_n)^\top$. Recall that $\mathcal{S}$ (without the subscript) denotes the whole process. We denote $\mathbf{\Sigma}_n$ as the covariance function of $\mathcal{S}_n$. In particular, $\gamma_{ij} = \mathrm{Cov}(S_i, S_j)$ is the (i, j)th entry of $\mathbf{\Sigma}_n$. Let $\boldsymbol{U}_{nk} = (\gamma_{1k}, \ldots, \gamma_{nk})^\top$. We also define $\tilde{\mathcal{S}}_n = (S_1 + \mu, S_2 + 2\mu, \ldots, S_n + n\mu)^\top$ as the mean-zero counterpart of $\mathcal{S}_n$. Let $M_n = \max_{1 \le l \le n} S_l$, i.e., the running-time maximum. Last, let $\mathbb{P}_n(\cdot) = \mathbb{P}(\cdot|\mathcal{S}_n)$.

2 Basic Strategy

In this section, we introduce the main idea of our algorithm. We start by introducing the milestone-event construction, which allows us to decompose the problem into generating a sequence of downward-crossing and upward-crossing events. We then discuss how to generate these events sequentially.

2.1 Milestone Events

An upward-crossing event is an event where the Gaussian process reaches a new maximum. Our goal is to find these upward-crossing events sequentially until we find the last one. To achieve this algorithmically, we also need to define some downward-crossing events.

Given $\mathcal{S}_n$, let

$$q(n) = \sum_{k=n+1}^{\infty} \mathbb{P}_n \left(S_k > M_n \right), \tag{2}$$

which provides an upper bound for the probability of having an upward-crossing event beyond n. We define a sequence of downward-crossing and upward-crossing event times as follows. Let $\tau_0^+ \equiv 0$. For $k \geq 1$, if $\tau_{k-1}^+ < \infty$, define

$$\tau_k^- := \widetilde{\min}\{n > \tau_{k-1}^+ : q(n) < 3/4\}, \quad \tau_k^+ := \inf\left\{ n > \tau_k^- : S_n > M_{\tau_k^-} \right\},$$

where $\widetilde{\min}$ is a notation we introduce to denote the caveat that we only require τ_k^- to be a time at which $q(n) < 3/4$. It does not need to be the first time at which this happens. This indicates that τ_k^- is not uniquely defined, which gives us some flexibility when designing the algorithm. For example, we can define τ_k^- as the first time after τ_{k-1}^+ where our algorithm can "detect" that $q(n) < 3/4$.

Note that the downward-crossing event time τ_k^- is defined in such a way that the chance of having an upward-crossing event after τ_k^- will be upper bounded by $3/4$, i.e., $\mathbb{P}_{\tau_k^-}(\tau_k^+ < \infty) \leq q(\tau_k^-) < 3/4$. This indicates that the upward-crossing event only happens a finite number of times. In particular, let $K = \sup\{k : \tau_k^+ < \infty\}$, i.e., the number of upward-crossing events. Then, K is stochastically bounded by a Geometric random variable with probability of success $1/4$.

Once we find K, $M_\infty = \max_{0 \leq n \leq \tau_{K+1}^-} S_n$. Thus, by generating the Gaussian process $\mathcal{S}$ up to the random time τ_{K+1}^-, we are able to recover the all-time maximum of $\mathcal{S}$. We also remark that finding K or τ_{K+1}^- is not straightforward since τ_{K+1}^- is not a stopping time, i.e., finding K requires knowing information of $\mathcal{S}$ beyond $\mathcal{S}_{\tau_K^+}$.

The next lemma shows that the downward-crossing events are well-defined.

Lemma 1 *For any fixed $a \in (0, 1)$, there exists a random integer L, such that for any $n > L$, $q(n) < a$. Moreover, for any $\eta > 0$, $\mathbb{E}[L^\eta] < \infty$.*

The proof of Lemma 1 and all the subsequent lemmas are delayed to the Appendix.

The next theorem shows that τ_{K+1}^- is well defined. In particular, even though τ_{K+1}^- is random, it has finite moments of all orders.

Theorem 1 *For any $\eta > 0$, $\mathbb{E}[(\tau_{K+1}^-)^\eta] < \infty$.*

The proof of Theorem 1 is based on constructing an algorithm to find τ_{K+1}^- and is provided in Sect. 4.2.1 as part of the complexity analysis of Algorithm 2.

2.2 Main Algorithm

Based on the milestone event construction, we next explain how to find these events sequentially.

Downward-crossing events. Finding the downward-crossing events can be done under the nominal measure. The main difficulty is to check whether $q(n) < 3/4$, because $q(n)$ involves an infinite sum which cannot be calculated exactly. To overcome the difficulty, we use a sandwiching construction. We derive a sequence of upper and lower bounds, $\{\mathcal{U}(\ell)\}_{\ell\geq 1}$ and $\{\mathcal{L}(\ell)\}_{\ell\geq 1}$ satisfying

$$\mathcal{L}(1) \leq \mathcal{L}(2) \leq \cdots \leq q(n) \leq \cdots \leq \mathcal{U}(2) \leq \mathcal{U}(1)$$

and $\lim_{\ell\to\infty} \mathcal{U}(\ell) = \lim_{\ell\to\infty} \mathcal{L}(\ell) = q(n)$. Then, we can sequentially refine the bounds until the stopping criterion—$\mathcal{U}(\ell) < 3/4$ or $\mathcal{L}(\ell) > 3/4$—-met to determine whether $q(n) < 3/4$. The technical and algorithmic details are provided in Sect. 3 and Algorithm 3.

Upward-crossing events. Sampling the upward-crossing event is more challenging. In particular, since $\mathbb{P}(\tau_k^+ < \infty | \mathcal{S}_{\tau_k^-}, \tau_k^-) < 1$, given τ_k^- and $\mathcal{S}_{\tau_k^-}$, if we generate the Gaussian process under the nominal measure until $S_n > M_{\tau_k^-}$, we may never be able to find τ_k^+, i.e., the algorithm can take an infinite amount of time. To overcome this challenge, we employ a rare-event simulation technique for Gaussian processes called TBS [12]. We remark that there are many candidate rare-event simulation techniques. We choose TBS because it is especially well-suited for Gaussian processes with general covariance structures. The implementation contains two key components: a change-of-measure and an acceptance-rejection step.

First, given $\mathcal{S}_n$, we introduce a new measure $\mathbb{Q}_n$ under which the upward-crossing event happens with probability 1. In particular, given $\mathcal{S}_n$, define

$$\kappa_n := \inf\{k > n : S_k > M_n\}.$$

Then, the new measure is defined through the following likelihood ratio

$$\frac{\mathrm{d}\mathbb{P}_n}{\mathrm{d}\mathbb{Q}_n} 1\{\kappa_n < \infty\} = \frac{\sum_{\ell=n+1}^{\infty} \mathbb{P}_n(S_\ell > M_n)}{\sum_{m=\kappa_n}^{\infty} \mathbb{P}_{\kappa_n}(S_m > M_n)}. \tag{3}$$

If we simulate $S_{n+1}, \ldots, S_{\kappa_n}$ under $\mathbb{Q}_n$, we have a proposed upward-crossing path. To algorithmically achieve this, we use TBS defined in Algorithm 1. The idea is to first sample a target upward-crossing time, and then use Gaussian bridge to sample the process conditional on the upward-crossing event at the target time.

Lemma 2 verifies that the probability measure induced by TBS is our target measure $\mathbb{Q}_n$. We comment that there remains an implementation challenge—how

to sample $N(n)$, since $q(n)$ cannot be evaluated exactly. We address this challenge using a sandwiching construction in Algorithm 4 in Sect. 3.

Lemma 2 *The probability measure induced by TBS satisfies* (3).

Second, we apply an acceptance-rejection step. In particular, given the proposed path $S_{n+1}, ..., S_{\kappa_n}$ under $\mathbb{Q}_n$, we sample a Bernoulli random variable I with probability of success

Algorithm 1: Target Bridge Sampling (Given $\mathcal{S}_n$)

1. Sample $N(n) > n$ with probability mass function

$$f_n(m) = \frac{\mathbb{P}_n(S_m > M_n)}{\sum_{k=n+1}^{\infty} \mathbb{P}_n(S_k > M_n)} = \frac{\mathbb{P}_n(S_m > M_n)}{q(n)}, \text{ for } m = n+1, \ldots. \tag{4}$$

2. Given $N(n) = m$ and $\mathcal{S}_n$, sample S_m conditional on $S_m \geq M_n$.
3. Conditional on S_m and $\mathcal{S}_n$, sample $S_{n+1}, ..., S_{m-1}$. Calculate $\kappa_n = \min\{k > n : S_k > M_n\}$.
4. Output $S_{n+1}, \ldots, S_{\kappa_n}$.

$$p(\kappa_n) = \left(\sum_{m=\kappa_n}^{\infty} \mathbb{P}_{\kappa_n}(S_m > M_n)\right) = \left(1 + \sum_{m=\kappa_n+1}^{\infty} \mathbb{P}_{\kappa_n}(S_m > M_n)\right)^{-1} \leq 1. \tag{5}$$

If the Bernoulli is a success, i.e., $I = 1$, the proposed path is accepted and it is the path leading to the next upward-crossing event as verified by the following lemma:

Lemma 3 *Given $\mathcal{S}_n$, for $S_{n+1}, ..., S_{\kappa_n}$ generated under $\mathbb{Q}_n$, let I denote a Bernoulli random variable with probability of success $p(\kappa_n)$. Then*

$$\mathbb{Q}_n(I = 1) = \frac{\mathbb{P}_n(\kappa_n < \infty)}{\sum_{l=n+1}^{\infty} \mathbb{P}_n(S_l > M_n)} = \frac{\mathbb{P}_n(\kappa_n < \infty)}{q(n)}$$

and $\mathbb{Q}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k | I = 1) = \mathbb{P}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k | \kappa_n < \infty)$.

We note from Lemma 3 that $\mathbb{Q}_n(I = 1) = \mathbb{P}_n(\kappa_n < \infty)/q(n) > \mathbb{P}_n(\kappa_n < \infty)$. Thus, given $\tau_k^- = n$ and $\mathcal{S}_n$, we generate another independent Bernoulli random variable J with probability of success $q(n)$. Note that

$$\mathbb{Q}_n(J = 1, I = 1) = q(n)\frac{\mathbb{P}_n(\kappa_n < \infty)}{q(n)} = \mathbb{P}_n(\kappa_n < \infty) = \mathbb{P}_n(\tau_k^+ < \infty),$$

$$\begin{aligned} &\mathbb{Q}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k | I = 1, J = 1) \\ &= \mathbb{Q}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k | I = 1) \text{ by independence} \\ &= \mathbb{P}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k | \kappa_n < \infty) \text{ by Lemma 3.} \end{aligned} \tag{6}$$

Meanwhile, we also note that

$$\mathbb{Q}_n(J=0)+\mathbb{Q}_n(J=1,I=0)=1-q(n)+q(n)\left(1-\frac{\mathbb{P}_n(\kappa_n<\infty)}{q(n)}\right) \\ =\mathbb{P}_n(\kappa_n=\infty)=\mathbb{P}_n(\tau_k^+=\infty). \tag{7}$$

This indicates that to determine whether $\tau_k^+ < \infty$, we first sample J. If $J = 0$, we can claim that $\tau_k^+ = \infty$. If $J = 1$, we further apply $\mathbb{Q}_n$ to sample a proposed path and sample I. If $I = 1$, we accept the proposed path as the path leading to the next upward-crossing event. If $I = 0$, we can claim that $\tau_k^+ = \infty$. We remark that sampling J and I is not straightforward, as $q(n)$ and $p(\kappa_n)$ can not be evaluated exactly. We will explain how to do so using a sandwiching construction in Algorithms 4 and 5 in Sect. 3.

We conclude the section by summarizing the ideas discussed above and presenting the main simulation algorithm—Algorithm 2.

Algorithm 2: Simulating the all-time maximum of $\mathcal{S}$

Step 0: Initialization.

1. Set $k = 0$, $\tau_k^+ = 0$, $n = 0$, and $S_0 = 0$.

Step 1: Downward-crossing event.

1. Sample S_{n+1} conditional on $\mathcal{S}_n$.
2. Call Algorithm 3 to sample $W \in \{0, 1\}$. If $W = 0$, set $n = n + 1$ and go back to **Step 1.1**. If $W = 1$, go to **Step 1.3**.
3. Set $n = \tau_k^-$ and $M_n = \max_{1\le l\le n} S_l$.

Step 2: Upward-crossing event.

1. Call Algorithm 4 to sample $J \sim$ Bernoulli$(q(n))$. If $J = 0$, go to **Step 3**. If $J = 1$, Algorithm 4 also outputs a random time $N \sim f_n(\cdot)$.
2. Given N, sample S_N according to $\mathbb{P}_n(S_N \in \cdot | S_N > M_n)$.
3. Conditional on S_N and $\mathcal{S}_n$, sample $S_{n+1}, ..., S_{N-1}$. Calculate $\kappa_n = \min\{l \ge n : S_l > M_n\}$.
4. Call Algorithm 5 to sample $I \sim$Bernoulli$(p(\kappa_n))$. If I=1, set k=$k + 1$, $\tau_k^+ = \kappa_n$, $n = \tau_k^+$ and go to **Step 1**. If $I = 0$, go to **Step 3**.

Step 3: Output.

1. Output $M_\infty = M_n$. (We can also output $\mathcal{S}_n$)

Theorem 2 *The output of Algorithm 2 follows the same distribution as* M_∞.

Before we prove Theorem 2, we need to introduce the details of the intermediate steps in Algorithm 2. Thus, the proof is delayed to Sect. 4.1.

3 Intermediate Steps in Algorithm 2

In this section, we present the details of the intermediate steps in Algorithm 2. In particular, we introduce Algorithms 3–5.

The fundamental challenge in these algorithms is that we need to compare a number, say u, to a probability p that cannot be evaluated exactly. We resolve this challenge by deriving a sequence of upper and lower bounds for p.

Given $\mathcal{S}_n$, recall from (2) that $q(n) = \sum_{k=n+1}^{\infty} \mathbb{P}_n(S_k > M_n)$. We define the lower bounds by simply truncating the infinite sum to a finite number of terms:

$$q(n, \ell) := \sum_{k=n+1}^{\ell} \mathbb{P}_n(S_k > M_n)$$

The upper bounds are more challenging to construct. Define

$$h(\ell) = \frac{8\sigma^2}{(1-H)\mu^2} \exp\left(-\frac{\mu^2}{16\sigma^2}\ell^{2-2H}\right), \tag{8}$$

$$B(n) = \max\left\{\left(\frac{2\sigma^2 n^H \|\mathbf{\Sigma}_n^{-1}\|_1 \|\tilde{\mathbf{S}}_n\|_1}{\mu}\right)^{\frac{1}{1-H}}, \left(\frac{2\sigma^2}{\pi\mu^2}\right)^{\frac{1}{2(1-H)}}, \left(\frac{2H-1}{1-H}\frac{16\sigma^2}{\mu^2}\right)^2, n+1\right\}. \tag{9}$$

Lemma 4 *For any $\eta > 0$, $\mathbb{E}[B(n)^\eta] < \infty$. For any $\ell \geq B(n)$,*

$$q(n, \ell) < q(n) \leq q(n, \ell) + h(\ell).$$

In addition, $q(n, \ell) \leq q(n, \ell+1) \leq \cdots \leq q(n) \leq \cdots \leq q(n, \ell+1) + h(\ell+1) \leq q(n, \ell) + h(\ell)$ and $\lim_{\ell\to\infty} q(n, \ell) = \lim_{\ell\to\infty} q(n, \ell) + h(\ell) = q(n)$.

Based on Lemma 4, we have constructed a proper sequence of lower and upper bounds for $q(n)$. These bounds allow us to check whether we have reached a downward-crossing event in Algorithm 3 and to sample $J \sim$Bernoulli$(q(n))$ in Algorithm 4.

Algorithm 3: Given n and $\mathcal{S}_n$, output W where $W = 1$ implies $q(n) < 3/4$.

0. Calculate $B(n)$ and set $\ell = \lceil B(n) \rceil$.
1. Sample $U_0 \sim$Uniform$[1/2, 3/4]$.
2. Calculate $\mathcal{L}(\ell) = q(n, \ell)$ and $\mathcal{U}(\ell) = q(n, \ell) + h(\ell)$.
3. If $U_0 \leq \mathcal{L}(\ell)$, set $W = 0$, and go to **Step 4**; if $U_0 \geq \mathcal{U}(\ell)$, set $W = 1$ and go to **Step 4**; otherwise, set $\ell = \ell + 1$ and go to **Step 2**.
4. Output W.

In Algorithm 3, we do not compare $q(n)$ to $3/4$ directly. Instead, we compare $q(n)$ to a uniform random variable on $[1/2, 3/4]$: U_0. In this case, $W = 1$ implies that $U_0 > q(n)$, which further implies that $q(n) < 3/4$, i.e., the criteria for the downward-crossing event is met.

Algorithm 4: Given n and $\mathcal{S}_n$, sample $J \sim$Bernoulli$(q(n))$.

0. Calculate $B(n)$ and set $\ell = \lceil B(n) \rceil$.
1. Sample $U \sim$Uniform[0, 1].
2. If $U \leq q(n, \ell)$, set $J = 1$ and $\ell = \min\{h \geq n+1 : q(n, h-1) < U \leq q(n, h)\}$, and go to **Step 5**; otherwise, set $\ell = \ell + 1$ and go to **Step 3**.
3. Calculate $\mathcal{L}(\ell) = q(n, \ell)$ and $\mathcal{U}(\ell) = q(n, \ell) + h(\ell)$.
4. If $U \leq \mathcal{L}(\ell)$, set J=1, N=ℓ, and go to **Step 5**; if $U \geq \mathcal{U}(\ell)$, set $J = 0$ and go to **Step 5**; otherwise, set $\ell = \ell + 1$ and go to **Step 3**.
5. If $J = 1$, output J and $N = \ell$; otherwise, output J

In Algorithm 4, when $J = 1$, it also outputs $N = \ell$. To see this, note that

$$\mathbb{P}_n(N = \ell | J = 1) = \frac{\mathbb{P}_n(\mathcal{L}(\ell-1) < U < \mathcal{L}(\ell))}{P(U < q(n))} = \frac{\mathbb{P}_n(S_\ell > M_n)}{\sum_{k=n+1}^{\infty} \mathbb{P}_n(S_k > M_n)} = f_n(\ell). \tag{10}$$

Thus, as a byproduct of Algorithm 4, we also get a sample of $N \sim f_n(\cdot)$.

Lastly, given κ_n and $\mathcal{S}_{\kappa_n}$, we develop an algorithm to sample Bernoulli$(p(\kappa_n))$ (Algorithm 5). Recall from (5) that $p(\kappa_n) = (\sum_{\ell=\kappa_n}^{\infty} \mathbb{P}_{\kappa_n}(S_\ell > M_n))^{-1}$. Define

$$\tilde{q}(n, \ell) := \sum_{i=k+1}^{\ell} \mathbb{P}_{\kappa_n}(S_i > M_n).$$

Then, we have the following analog of Lemma 4.

Lemma 5 *For $\ell \geq B(\kappa_n)$, where $B(k)$ is defined in (9),*

$$(1 + \tilde{q}(n, \ell) + h(\ell))^{-1} \leq p(\kappa_n) \leq (1 + \tilde{q}(n, \ell))^{-1},$$

where $h(\ell)$ is defined in (8). In addition, $(1 + \tilde{q}(n, \ell) + h(\ell))^{-1} \leq (1 + \tilde{q}(n, \ell + 1) + h(\ell + 1))^{-1} \leq \cdots \leq p(\kappa_n) \leq \cdots \leq (1 + \tilde{q}(n, \ell + 1))^{-1} \leq (1 + \tilde{q}(n, \ell))^{-1}$ and $\lim_{\ell \to \infty}(1 + \tilde{q}(n, \ell))^{-1} = \lim_{\ell \to \infty}(1 + \tilde{q}(n, \ell) + h(\ell))^{-1} = p(\kappa_n)$.

Then, Algorithm 5 follows based on the sequence of bounds in Lemma 5.

Algorithm 5: Given κ_n and $\mathcal{S}_{\kappa_n}$, sample $I \sim$Bernoulli$(p(\kappa_n))$.

0. Calculate $B(\kappa_n)$ and set $\ell = \lceil B(\kappa_n) \rceil$.
1. Sample $U \sim$Uniform$[0, 1]$.
2. Calculate $\mathcal{L}(\ell) = (1 + \tilde{q}(n, \ell) + h(\ell))^{-1}$ and $\mathcal{U}(\ell) = (1 + \tilde{q}(n, \ell))^{-1}$.
3. If $U \leq \mathcal{L}(\ell)$, set $I = 1$ and go to **Step 4**; if $U \geq \mathcal{U}(\ell)$, set $I = 0$ and go to **Step 4**; otherwise, set $\ell = \ell + 1$ and go to **Step 2**.
4. Output I

4 Analysis of Algorithm 2

In this section, we provide detailed analysis to verify the correctness and complexity of Algorithm 2. In particular, we provide the proof of Theorems 1 and 2.

4.1 Output Analysis

Proof *(of Theorem 2)* In Step 1 of Algorithm 2, we simulate $\mathcal{S}$ under the nominal measure.

For Step 2, we first verify the output of Algorithms 4. Because $\lim_{\ell\to\infty} q(n, \ell) + h(\ell) = q(n)$ by Lemma 4, we have $\mathbb{P}_n(J = 1) = \mathbb{P}_n(U < q(n)) = q(n)$. In addition, from (10), we have $\mathbb{P}_n(N = \ell | J = 1) = f_n(\ell)$. Therefore, in Step 2.1, if $J = 1$, $N \sim f_n(\cdot)$. Then, Steps 2.1–2.3 constitute the TBS procedure, i.e., $S_{n+1}, \ldots, S_{\kappa_n}$ in Step 2.3 is a sample path drawn under $\mathbb{Q}_n$. We next verify the output of Algorithm 5. Since $\lim_{\ell\to\infty}(1 + \tilde{q}(n, \ell) + h(\ell))^{-1} = p(\kappa_n)$ by Lemma 5, $\mathbb{Q}_n(\ell = 1) = \mathbb{Q}_n(U < p(\kappa_n)) = p(\kappa_n)$. Therefore, Step 2.4 is the acceptance-rejection step. From (6),

$$\mathbb{Q}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k | I = 1, J = 1) = \mathbb{P}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k | \kappa_n < \infty).$$

Thus, if we accept the path, it is is the path leading to the next upward-crossing event. Meanwhile, from (7), $\mathbb{Q}_n(J = 0) + \mathbb{Q}_n(J = 1, I = 0) = \mathbb{P}_n(\tau_k^+ = \infty)$. Thus, if we go to Step 3, there is no more record breakers, i.e., $n = \kappa_{K+1}^-$. □

4.2 Complexity Analysis

In this section, we conduct detailed complexity analysis of Algorithm 2. Note that the computational cost of Algorithm 2 is random due the random length of the sample path, i.e., τ_{K+1}^-, and the random number of iterations in Algorithms 3–5. We will show that these random quantities have finite moments of all orders.

Let $\mathcal{C}(i)$ denote the computational complexity of generating the i-th sample of M_∞ from Algorithm 2, $M_\infty(i)$. Let $\mathcal{N}(c)$ denote the total number of $M_\infty(i)$'s generated with a computational budget of c, i.e., $\mathcal{N}(c) := \max\{n \geq 0 : \mathcal{C}(1) + \cdots + \mathcal{C}(n) \leq c\}$. Then, if $\mathbb{E}[\mathcal{C}(1)] < \infty$, which is the case in our setting, and $\mathrm{Var}(M_\infty(1)) < \infty$, we achieve the canonical $\sqrt{c}$ rate of convergence [26], i.e.,

$$\sqrt{c}\left(\sum_{i=1}^{\mathcal{N}(c)} M_\infty(i) - \mathbb{E}[M_\infty(1)]\right) \Rightarrow \sqrt{\mathrm{e}[\mathcal{C}(1)]\mathrm{Var}(M_\infty(1))}\boldsymbol{N}(0,1) \mathrm{as} c \to \infty,$$

where $\boldsymbol{N}(0, 1)$ denote a Gaussian distribution with mean 0 and variance 1. It is also important to study how the complexity, e.g., $\mathbb{E}[\mathcal{C}(1)]$, depends on the covariance structure. We investigate this through numerical experiments in Sect. 5.

4.2.1 Proof of Theorem 1

Before we prove Theorem 1, we first introduce an auxiliary lemma.

Lemma 6 *Given $\mathcal{S}_n$, for $N(n)$ and J generated in Step 2.1 in Algorithm 2, we have for any $\eta > 0$, $\mathbb{E}[N(n)^\eta 1\{J = 1\}|\mathcal{S}_n] < \infty$.*

Proof (*of Theorem 1*) Let $\tilde{T} = \tau^-_{K+1}$, which is the length of the Gaussian process generated in Algorithm 2.

First, let G denote number of times we visit Step 1 in Algorithm 2, which is the same as the number of times we visit Step 2. We note that when we visit Step 2.1 in Algorithm 2, if $J = 0$, the algorithm terminates. By the construction of downward-crossing event, $q(n) < 3/4$ when in Step 2.1. Thus, G is stochastically upper bounded by a Geometric random variable with probability of success $1/4$.

Second, we study the number of elements of the Gaussian process generated in Step 1. If we set $a = 1/2$ in Lemma 1, then there exists a random time, $\tilde{L}$, which has finite moments of all orders, such that for $n > \tilde{L}$, $q(n) < 1/2$. When $q(n) < 1/2$, $W = 1$ in Algorithm 3 with probability 1. This suggests that when $n > \tilde{L}$, every time we go to Step 1, we only generate one more point of the Gaussian process, i.e., $\tau^-_{k+1} = \tau^+_k + 1$. Thus, the number of elements of the Gaussian process we generate in Step 1 is upper bounded by $\tilde{L} + G$, which has finite moments of all orders.

Third, we study the number of elements of the Gaussian process we generate in Step 2 in Algorithm 2. When we visit Step 2 at $\tau^-_k = n$, if $J = 1$ in Step 2.1, we generate a path leading to the next upward-crossing event. The number of elements of the Gaussian process we generate is upper bounded by $N(n) - n$. By Lemma 6, $N(n)$ has finite moments of all orders. Then, the number of elements of the Gaussian process we generate in Step 2 is upper bounded by $\sum_{k=1}^{G} N(\kappa^-_k) - \kappa^-_k$, which has finite moments of all orders. □

4.2.2 Complexity of Algorithms 3–5

Based on the sandwiching construction, in Algorithms 3–5 we compare a uniform random variable with an unknown probability p using the iteratively updated bounds for p. We present the analysis for Algorithm 4 next.

Given $\mathcal{S}_n$, let $\Theta - n$ denote the number of iterations in Algorithm 4:

$$\Theta := \inf\{l > n : U \leq q(n,l) \text{ or } U \geq q(n,l) + h(l)\},$$

where $U \sim$Uniform[0, 1]. Then, for any $l > B(n)$,

$$\mathbb{P}_n(\Theta > l) = \mathbb{P}_n(q(n,l) < U < q(n,l) + h(l)) = h(l).$$

We first note that for any $\eta > 0$, $\sum_{l=1}^{\infty} l^{\eta} h(l) < \infty$. Next, from Lemma 4, $\mathbb{E}[B(n)^{\eta}] < \infty$. Thus, Θ has finite moments of all orders.

Similarly, we can show that the number of iterations in Algorithm 3 and 5 also has finite moments of all orders. Note that for the sandwiching construction in Algorithm 5, we have $(1 + q_b(n,l))^{-1} - (1 + q_b(n,l) + h(l))^{-1} \leq h(l)$.

5 Numerical Experiments

In this section, we implement our algorithm and test its performance based on fractional Gaussian noises. This complements our complexity analysis in Sect. 4.2. Consider $S_n = -n\mu + B^H(n)$ where B^H is an fBM with Hurst index $H \in (0, 1)$. We set $\mu = 1$ and use three different values of H: $H = 0.3, 0.5$, and 0.7, corresponding to the cases where X_n's are negatively , un-, and positively correlated, respectively.

In Fig. 1, we show the distribution of M_∞ based on 10^3 independent copies of it, i.e. we repeat Algorithm 2 10^3 times for each model. On the left panel, we plot the histogram of M_∞. We note that there is a very high probability that $M_\infty = 0$. On the right panel, we plot the histogram of M_∞ conditional on $M_\infty > 0$. We observe that as H increases, the tail of the distribution of M_∞ becomes heavier. In particular, M_∞ is more likely to take very large values when $H = 0.7$ than when $H = 0.3$.

We next look at the complexity of our algorithm. We analyze two quantities: 1) the length of the sample path generated in Algorithm 2, which we denote as $\tilde{T}$ and 2) the number of iterations in Algorithm 4, which we denote as Θ_m. We note that even though $\tilde{T}$ is a natural measure of the complexity, in actual implementations, the most time-consuming part is the intermediate step—Algorithm 4, i.e., sampling a Bernoulli random variable with probability of success $q(n)$. In Fig. 2, we plot the histogram of $\tilde{T}$ (left panel) and Θ_m (right panel) based on 10^3 independent replications of Algorithm 2. We observe that as H increases, $\tilde{T}$ tends to take larger values. More importantly, as H increases, the tail of the distribution of Θ_m becomes heavier very rapidly. For example, when $H = 0.3$, Θ_m tends to take very small values, i.e., less

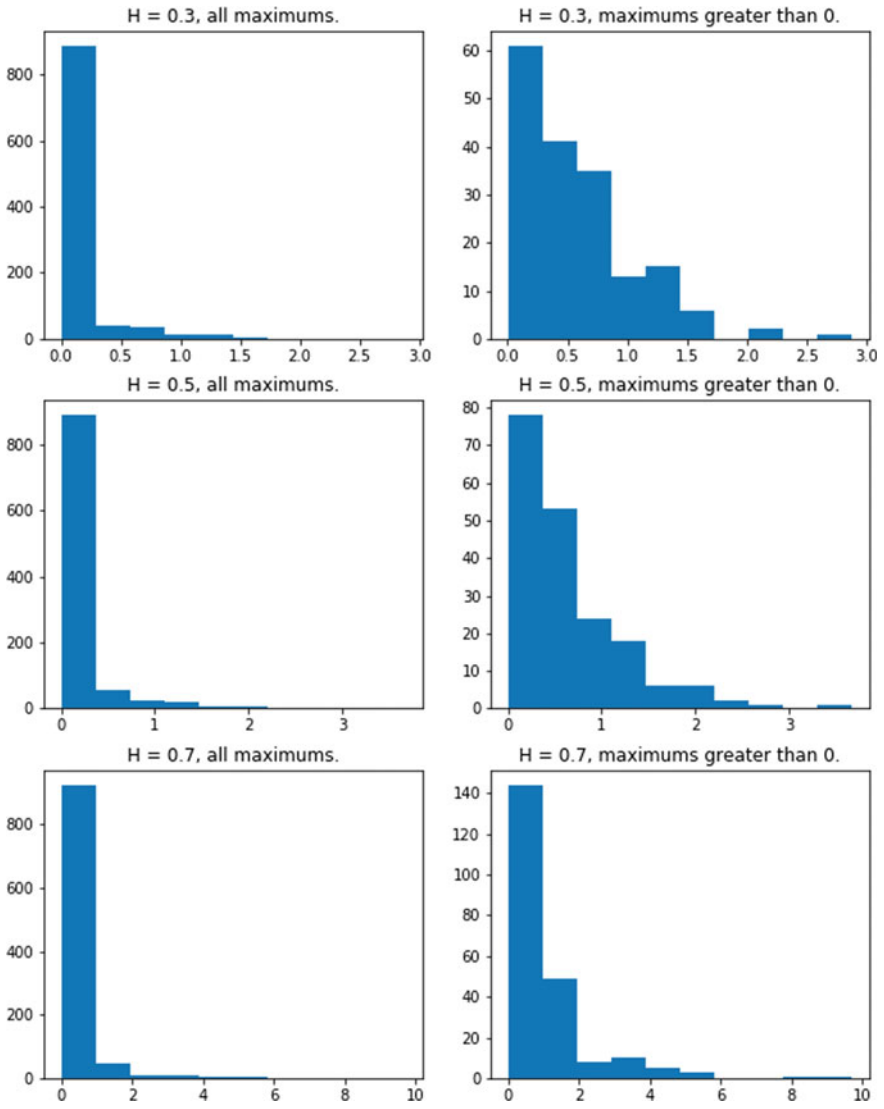


Fig. 1 Histograms of M_∞ where $S_n = -n + B^H(n)$ and $H = 0.3, 0.5$, or 0.7

than 14. However, when $H = 0.7$, the distribution of Θ_m has an extremely heavy tail. In our sample, Θ_m can be as large as 5×10^5.

Based on the numerical experiments, we note that as H increase, the computational complexity increases. Indeed, when $H \geq 0.9$ in our fractional Gaussian noise example, Θ_m in Algorithm 4 can be too large to handle computationally, e.g, 10^{17}. Note that for $\ell > B(n)$,

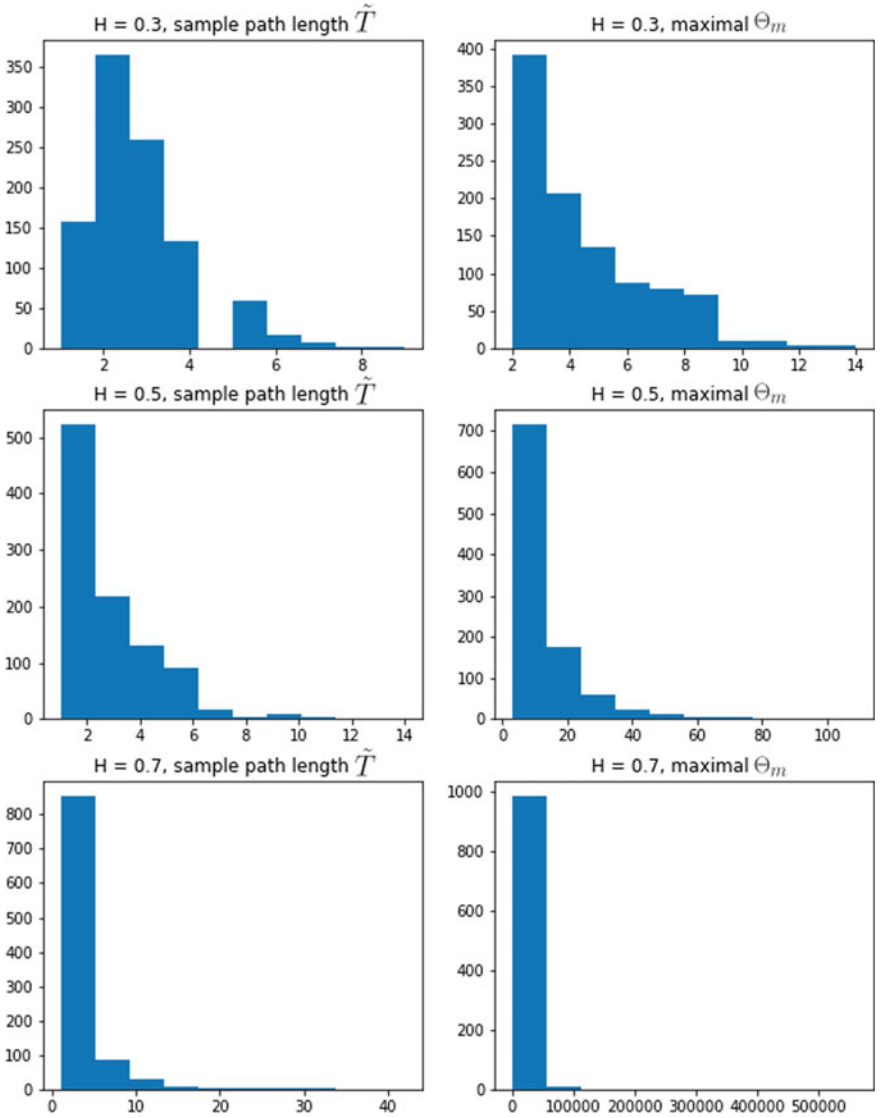


Fig. 2 Histograms of $\tilde{T}$ and Θ_m. $S_n = -n + B^H(n)$ and $H = 0.3, 0.5$, or 0.7

$$\mathbb{P}_n(\Theta_m > \ell) = h(\ell) = \frac{8\sigma^2}{(1-H)\mu^2} \exp\left(-\frac{\mu^2}{16\sigma^2}\ell^{2-2H}\right),$$

which can decay very slowly when H is close to 1. Even $B(n)$ (defined in (9)) can be very large in this case. How to sample M_∞ when H is close to 1 in a computationally efficient way would require fundamentally new developments, which is an interesting future research direction.

6 Conclusion

In this paper, we develop an exact sampling algorithm for the all the maximum of negative-drifted Gaussian processes with general covariance structures. The complexity of our algorithm is random but has finite moments of all orders. Our developments involve novel applications of several simulation techniques, including the milestone-event construction, a rare event simulation technique called TBS, and the sandwiching construction. We test the performance of algorithm numerically and discuss limitations in implementation when H is close to 1.

Appendix

Proof of Lemma 1

Proof Note that S_k conditional on $\mathcal{S}_n$ is still a Gaussian random variable with conditional mean

$$\mu_n(k) = \mathbb{E}[S_k|\mathcal{S}_n] = -k\mu + \boldsymbol{U}_{nk}^\top \boldsymbol{\Sigma}_n^{-1} \tilde{\mathcal{S}}_n,$$

and conditional variance

$$\sigma_n(k)^2 = \mathrm{Var}[S_k|\mathcal{S}_n] = \sigma^2 k^{2H} - \boldsymbol{U}_{nk}^\top \boldsymbol{\Sigma}_n^{-1} \boldsymbol{U}_{nk}.$$

The proof of the lemma is divided into three steps. We first establish bounds for the conditional mean $\mu_n(k)$. Let $\tilde{\mu}_n(k) = \boldsymbol{U}_{nk}^\top \boldsymbol{\Sigma}_n^{-1} \tilde{\mathcal{S}}_n$. As $\tilde{\mu}_n(k)$ is a linear combination of $\tilde{\mathcal{S}}_n$, it follows a Normal distribution with mean 0 and variance $\boldsymbol{U}_{nk}^\top \boldsymbol{\Sigma}_n^{-1} \boldsymbol{U}_{nk}$. By the law of total variance, $\boldsymbol{U}_{nk}^\top \boldsymbol{\Sigma}_n^{-1} \boldsymbol{U}_{nk} < \sigma^2 k^{2H}$. In this case, for any fixed $\delta \in (0, \mu)$,

$$\mathbb{P}(\tilde{\mu}_n(k) > \delta k) \le \mathbb{P}\left(\frac{\tilde{\mu}_n(k)}{\sqrt{\boldsymbol{U}_{nk}^\top \boldsymbol{\Sigma}_n^{-1} \boldsymbol{U}_{nk}}} > \frac{\delta k}{\sigma k^H}\right) = \bar{\Phi}\left(\frac{\delta}{\sigma} k^{1-H}\right). \tag{11}$$

Then,

$$\sum_{n=1}^{\infty}\sum_{k=n}^{\infty}\mathbb{P}(\tilde{\mu}_n(k) > \delta k) = \sum_{k=1}^{\infty}\sum_{n=1}^{k}\mathbb{P}(\tilde{\mu}_n(k) > \delta k) \le \sum_{k=1}^{\infty} k\bar{\Phi}\left(\frac{\delta}{\sigma}k^{1-H}\right) < \infty.$$

By Borel-Cantelli Lemma, there exits a random number $L_0 \ge n$, which is finite almost surely, such that when $k > L_0$, $\tilde{\mu}_n(k) \le \delta k$, which further implies that $\mu_n(k) \le -(\mu - \delta)k$.

We next establish bounds for $\sum_{n=1}^{\infty} q(n)$. For $k > L_0$, we have $\mu_n(k) \le -(\mu - \delta)k$ and $\sigma_n(k)^2 \le \sigma^2 k^{2H}$. Thus, for any $b \ge 0$,

$$\mathbb{P}_n(S_k > b) \le \mathbb{P}_n\left(\frac{S_k - \mu_n(k)}{\sigma_n(k)} > \frac{b + (\mu - \delta)k}{\sigma k^H}\right) \le \bar{\Phi}\left(\frac{\mu - \delta}{\sigma}k^{1-H}\right). \quad (12)$$

Based on the analysis above, let $b = \max_{1\le l\le n} S_l$. We decompose $\sum_{n=1}^{\infty} q(n)$ into three parts:

$$\sum_{n=1}^{\infty}\sum_{k=n}^{\infty}\mathbb{P}_n(S_k > b) \le \underbrace{\sum_{n=1}^{L_0}\sum_{k=n}^{L_0}\mathbb{P}_n(S_k > b)}_{\text{(I)}} + \underbrace{\sum_{n=1}^{L_0}\sum_{k=L_0}^{\infty}\mathbb{P}_n(S_k > b)}_{\text{(II)}} + \underbrace{\sum_{n=L_0}^{\infty}\sum_{k=n}^{\infty}\mathbb{P}_n(S_k > b)}_{\text{(III)}}.$$

Part (I) only involves a finite number of terms. For part (II), from (12), we have

$$\text{(II)} \le L_0\sum_{k=L_0}^{\infty}\bar{\Phi}\left(\frac{\mu - \delta}{\sigma}k^{1-H}\right) < \infty.$$

Similarly, for part (III), from (12), we have

$$\text{(III)} = \sum_{k=L_0}^{\infty}\sum_{n=L_0}^{k}\mathbb{P}_n(S_k > b) \le \sum_{k=L_0}^{\infty}(k - L_0)\bar{\Phi}\left(\frac{\mu - \delta}{\sigma}k^{1-H}\right) < \infty.$$

Putting parts (I)–(III) together, we have $\sum_{n=1}^{\infty} q(n) < \infty$. By Borell-Cantelli Lemma, there exits L, which is finite almost surely, such that for any $n > L$, $q(n) < a$.

Lastly, we show that $\mathbb{E}[L^\eta] < \infty$ for any $\eta > 0$. Let L_1 denote a large enough constant, such that $\sum_{k=L_1}^{\infty}\bar{\Phi}\left(\frac{\mu-\delta}{\sigma}k^{1-H}\right) < a$. Then, $L \le \max\{L_0, L_1\}$. Thus, to prove $\mathbb{E}[L^\eta] < \infty$, we only need to show that $\mathbb{E}[L_0^\eta] < \infty$. Define $\mathcal{A}_n = \bigcup_{k=n}^{\infty}\{\tilde{\mu}_n(k) > \delta k\}$. Then $L_0^\eta \le \sum_{n=1}^{\infty} 1\{\mathcal{A}_n\}n^\eta$, and

$$\begin{aligned}\mathbb{E}[L_0^\eta] \le \mathbb{E}\left[\sum_{n=1}^{\infty} 1\{\mathcal{A}_n\}n^\eta\right] &= \sum_{n=1}^{\infty}\sum_{k=n}^{\infty}\mathbb{P}(\tilde{\mu}_n(k) > \delta k)n^\eta \\ &= \sum_{k=1}^{\infty}\sum_{n=1}^{k} n^\eta\mathbb{P}(\tilde{\mu}_n(k) > \delta k) \le \sum_{k=1}^{\infty} k^\eta\bar{\Phi}\left(\frac{\delta}{\sigma}k^{1-H}\right) < \infty,\end{aligned}$$

where the last inequality follows from (11). □

Proof of Lemma 2

Proof With a little abuse of notation, we denote $\mathbb{Q}_n$ as the measure induced by the TBS procedure. First note that

$$\begin{aligned}
&\mathbb{Q}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k) = \sum_{m=n+1}^{\infty} f_n(m)\mathbb{P}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k | S_m > b) \\
&= \sum_{m=n+1}^{\infty} f_n(m) \frac{\mathbb{P}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k, S_m > b)}{\mathbb{P}_n(S_m > b)} \\
&= \sum_{m=n+1}^{\infty} \frac{\mathbb{P}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k, S_m > b)}{\sum_{\ell=n+1}^{\infty} \mathbb{P}_n(S_\ell > b)} \\
&= \sum_{m=n+1}^{\infty} \mathbb{P}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k) \frac{\mathbb{P}_n(S_m > b | (S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k)}{\sum_{\ell=n+1}^{\infty} \mathbb{P}_n(S_\ell > b)} \\
&= \mathbb{P}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k) \frac{\sum_{m=k}^{\infty} \mathbb{P}_n(S_m > b | (S_{n+1}, ..., S_k) \in \cdot, \tau(b) = k)}{\sum_{\ell=n+1}^{\infty} \mathbb{P}_n(S_\ell > b)}.
\end{aligned}$$

Thus, $\frac{\mathrm{d}\mathbb{P}_n}{\mathrm{d}\mathbb{Q}_n}(S_{n+1}, ..., S_{\kappa_n}, \kappa_n < \infty) = \frac{\sum_{\ell=n+1}^{\infty} \mathbb{P}_n(S_\ell > b)}{\sum_{m=\kappa_n}^{\infty} \mathbb{P}_{\kappa_n}(S_m > b)}$. □

Proof of Lemma 3

Proof Let $\mathbb{E}_{\mathbb{Q}}$ denote the expectation under measure $\mathbb{Q}$. Suppose $M_n = b$. First note that by Lemma 2,

$$\begin{aligned}
\mathbb{Q}_n(I = 1) &= \mathbb{E}_{\mathbb{Q}_n}\left[\left(\sum_{\ell=\kappa_n}^{\infty} \mathbb{P}_{\kappa_n}(S_\ell > b)\right)^{-1}\right] \\
&= \mathbb{E}_{\mathbb{P}_n}\left[\frac{1}{\sum_{\ell=\kappa_n}^{\infty} \mathbb{P}_{\kappa_n}(S_\ell > b)} \frac{\sum_{\ell=\kappa_n}^{\infty} \mathbb{P}_{\kappa_n}(S_\ell > b)}{\sum_{\ell=n+1}^{\infty} \mathbb{P}_n(S_\ell > b)} 1\{\kappa_n < \infty\}\right] \\
&= \mathbb{E}_{\mathbb{P}_n}\left[\left(\sum_{\ell=n+1}^{\infty} \mathbb{P}_n(S_\ell > b)\right)^{-1} 1\{\kappa_n < \infty\}\right] = \frac{\mathbb{P}_n(\kappa_n < \infty)}{\sum_{\ell=n+1}^{\infty} \mathbb{P}_n(S_\ell > b)}
\end{aligned} \tag{13}$$

Next, by Bayes rule,

$$\mathbb{Q}_n((S_{n+1}, ..., S_{\kappa_n}) \in \cdot, \kappa_n \in \cdot | I = 1) = \frac{\mathbb{Q}_n(I = 1 | \kappa_n, \mathcal{S}_{\kappa_n}) \mathbb{Q}_n((S_{n+1}, ..., S_{\kappa_n}) \in \cdot, \kappa_n \in \cdot)}{\mathbb{Q}_n(I = 1)}. \tag{14}$$

As $\mathbb{Q}_n(I = 1 | \kappa_n, (S_{n+1}, ..., S_{\kappa_n})) = \frac{1}{\sum_{\ell=\tau(b)}^{\infty} \mathbb{P}_{\kappa_n}(S_\ell > b)}$, plugging (13) in (14), we have

$$
\begin{aligned}
&\mathbb{Q}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k | I = 1) \\
&= \frac{1}{\sum_{\ell=k}^{\infty} \mathbb{P}_k(S_\ell > b)} \mathbb{Q}_n((S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k) \frac{\sum_{\ell=n+1}^{\infty} \mathbb{P}_n(S_\ell > b)}{\mathbb{P}_n(\kappa_n < \infty)} \\
&= \mathbb{E}_{\mathbb{Q}_n} \left[1\{(S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k\} \frac{\sum_{\ell=n+1}^{\infty} \mathbb{P}_n(S_\ell > b)}{\sum_{\ell=k}^{\infty} \mathbb{P}_k(S_\ell > b)} \right] \frac{1}{\mathbb{P}_n(\kappa_n < \infty)} \\
&= \frac{\mathbb{E}_{\mathbb{P}_n} \left[1\{(S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k\} \right]}{\mathbb{P}_n(\kappa_n < \infty)} \text{ by Lemma 2} \\
&= \mathbb{P}_n(S_{n+1}, ..., S_k) \in \cdot, \kappa_n = k | \kappa_n < \infty).
\end{aligned}
$$

□

Proof of Lemma 4

Proof Given $\mathcal{S}_n$, suppose $M_n = b$. We also define

$$
N_1 = \left(\frac{2\sigma^2 n^H \|\boldsymbol{\Sigma}_n^{-1}\|_1 \|\tilde{\mathbf{S}}_n\|_1}{\mu} \right)^{\frac{1}{1-H}}, \; N_2 = \left(\frac{2\sigma^2}{\pi \mu^2} \right)^{\frac{1}{2(1-H)}}, \text{ and } N_3 = \left(\frac{2H-1}{1-H} \frac{16\sigma^2}{\mu^2} \right)^2.
$$

Note that for any $k > n$, S_k conditional on $\mathcal{S}_n$ is still a Gaussian random variable with conditional mean $\mu_n(k) = \mathbb{E}[S_k|\mathcal{S}_n] = -k\mu + \boldsymbol{U}_{nk}^\top \boldsymbol{\Sigma}_n^{-1} \tilde{\mathcal{S}}_n$, and conditional variance $\sigma_n(k)^2 = \text{Var}[S_k|\mathcal{S}_n] = \sigma^2 k^{2H} - \boldsymbol{U}_{nk}^\top \boldsymbol{\Sigma}_n^{-1} \boldsymbol{U}_{nk}$.

We first establish the sequence of bounds. The lower bound is straightforward. For the upper bound, note that for $k \geq N_1$,

$$
\mu_n(k) \leq -k\mu + \sigma^2 (nk)^H \|\boldsymbol{\Sigma}_n^{-1} \tilde{\mathcal{S}}_n\|_1 \leq -k\mu + \sigma^2 (nk)^H \|\boldsymbol{\Sigma}_n^{-1}\|_1 \|\tilde{\mathcal{S}}_n\|_1 \leq -\frac{k\mu}{2}.
$$

Next, note that for $k \geq \max\{N_1, N_2\}$,

$$
\mathbb{P}_n(S_k > b) \leq \frac{1}{\sqrt{2\pi}} \frac{\sigma_n(k)}{b - \mu_n(k)} \exp\left(-\frac{(b - \mu_n(k))^2}{\sigma_n(k)^2} \right) \leq \exp\left(-\frac{\mu^2}{8\sigma^2} k^{2-2H} \right). \tag{15}
$$

To see the second inequality, note that when $k \geq N_1$, $\mu_n(k) \leq -k\mu/2$ and $\sigma_n(k) \leq \sigma k^H$. Thus, $\frac{b-\mu_n(k)}{\sigma_n(k)} \geq \frac{b+k\mu}{2\sigma k^H} \geq \frac{\mu}{2\sigma} k^{1-H}$. And for $k \geq N_2$, $\frac{1}{\sqrt{2\pi}} \left(\frac{\mu}{2\sigma} k^{1-H} \right)^{-1} \leq 1$.

Lastly, we have for $\ell \geq \max\{N_1, N_2, N_3\}$,

$$\begin{aligned}
\sum_{k=\ell+1}^{\infty} \mathbb{P}_n(S_k > b) &\le \sum_{k=\ell+1}^{\infty} \exp\left(-\frac{\mu^2}{8\sigma^2}k^{2-2H}\right) \quad \text{from 15 as } \ell \ge \max\{N_1, N_2\} \\
&\le \int_{\ell}^{\infty} \exp\left(-\frac{\mu^2}{8\sigma^2}k^{2-2H}\right) \mathrm{d}k \\
&= \frac{1}{2-2H} \int_{\ell^{2-2H}}^{\infty} y^{(2H-1)/(2-2H)} \exp\left(-\frac{\mu^2}{8\sigma^2}y\right) \mathrm{d}y \\
&\le \frac{1}{2-2H} \int_{\ell^{2-2H}}^{\infty} \exp\left(-\frac{\mu^2}{16\sigma^2}y\right) \mathrm{d}y \quad \text{as } \ell \ge N_3 \\
&\le \frac{8\sigma^2}{(1-H)\mu^2} \exp\left(-\frac{\mu^2}{16\sigma^2}\ell^{2-2H}\right) = h(\ell).
\end{aligned}$$

For $\mathbb{E}[B(n)^\eta]$, we first note that N_2 and N_3 are finite constants. Thus, we only need to show that $\mathbb{E}[N_1^\eta] < \infty$. For any fixed n,

$$\begin{aligned}
\mathbb{E}[N_1^\eta] &= \mathbb{E}\left[\left(\frac{2\sigma^2 n^H \|\mathbf{\Sigma}_n^{-1}\|_1 \|\tilde{\boldsymbol{S}}_n\|_1}{\mu}\right)^{\frac{\eta}{1-H}}\right] \\
&= \left(\frac{2\sigma^2 n^H \|\mathbf{\Sigma}_n^{-1}\|_1}{\mu}\right)^{\frac{\eta}{1-H}} \mathbb{E}\left[\left(\sum_{k=1}^{n} |S_k + k\mu|\right)^{\frac{\eta}{1-H}}\right] \\
&\le \left(\frac{2\sigma^2 n^H \|\mathbf{\Sigma}_n^{-1}\|_1}{\mu}\right)^{\frac{\eta}{1-H}} n^{\frac{\eta}{1-H}-1} \sum_{k=1}^{n} \mathbb{E}\left[|S_k + k\mu|^{\frac{\eta}{1-H}}\right] \\
&= \left(\frac{2\sigma^2 \|\mathbf{\Sigma}_n^{-1}\|_1}{\mu}\right)^{\frac{\eta}{1-H}} n^{\frac{\eta H+\eta+H-1}{1-H}} \frac{\Gamma(\frac{\eta/(1-H)+1}{2})}{\sqrt{\pi}} (2\sigma^2)^{\frac{\eta}{2(1-H)}} \sum_{k=1}^{n} k^{\frac{\eta H}{1-H}} \\
&\le \left(\frac{2^{3/2}\sigma^3 \|\mathbf{\Sigma}_n^{-1}\|_1}{\mu}\right)^{\frac{\eta}{1-H}} \frac{\Gamma(\frac{\eta/(1-H)+1}{2})}{\sqrt{\pi}} n^{\frac{2\eta H+\eta}{1-H}}.
\end{aligned}$$

□

Proof of Lemma 5

Proof Given κ_n and $\mathcal{S}_{\kappa_n}$, suppose $M_n = b$. First note that

$$\tilde{q}(n, \ell) \le \tilde{q}(n, \ell) \le \cdots \le \sum_{i=\kappa_n+1}^{\infty} \mathbb{P}_{\kappa_n}(S_i > b).$$

Next, following the proof of Lemma 4, we have for $\ell \geq B(\kappa_n)$,

$$\tilde{q}(n,\ell) + h(\ell) \geq \tilde{q}(n,\ell+1) + h(\ell+1) \geq \cdots \geq \sum_{i=\kappa_n+1}^{\infty} \mathbb{P}_{\kappa_n}(S_i > b).$$

Since $\mathbb{P}_k(S_k > b) = 1$, $p(k) = (1 + \sum_{i=k+1}^{\infty} \mathbb{P}_k(S_i > b))^{-1}$, and for $\ell \geq B(\kappa_n)$, $(1 + \tilde{q}(n,\ell) + h(\ell))^{-1} \leq p(\kappa_n) \leq (1 + \tilde{q}(n,\ell))^{-1}$. The rest of the results follow similarly. □

Proof of Lemma 6

Proof We first note that in Step 2.1 in Algorithm 2, $\mathbb{P}_n(N(n) = \ell, J = 1) = \mathbb{P}_n(S_\ell > M_n)$. Next, following the same lines of analysis as the proof of Lemma 1, we have for any $\delta > 0$, there exists $L_0 > 0$ such that for $\ell > L_0$, $\mathbb{P}_n(S_\ell > M_n) \leq \bar{\Phi}\left(\frac{\mu-\delta}{\sigma}\ell^{1-H}\right)$, and for any $\eta > 0$, $\mathbb{E}[L_0^\eta] < \infty$. Then for any $\eta > 0$,

$$\begin{aligned}\mathbb{E}[N(n)^\eta | \mathcal{S}_n] &= \sum_{\ell=n+1}^{\infty} \ell^\eta \mathbb{P}_n(N(n) = \ell) \\ &\leq \mathbb{E}[N_0^\eta | \mathcal{S}_n] + \mathbb{E}\left[\sum_{\ell=N_0+1}^{\infty} \ell^\eta \bar{\Phi}\left(\frac{\mu-\delta}{\sigma}\ell^{1-H}\right) \middle| \mathcal{S}_n\right].\end{aligned}$$

Thus,

$$\mathbb{E}[N(n)^\eta] = \mathbb{E}[\mathbb{E}[N(n)^\eta | \mathcal{S}_n]] \leq \mathbb{E}[N_0^\eta] + \mathbb{E}\left[\sum_{\ell=N_0+1}^{\infty} \ell^\eta \bar{\Phi}\left(\frac{\mu-\delta}{\sigma}\ell^{1-H}\right)\right] < \infty.$$

□

References

1. Addie, R., Mannersalo, P., Norros, I.: Most probable paths and performance formulae for buffers with Gaussian input traffic. Eur. Trans. Telecommun. **13**(3), 183–196 (2002)
2. Adler, R.J., Blanchet, J.H., Liu, J.: Efficient Monte Carlo for high excursions of Gaussian random fields. Ann. Appl. Probab. **22**(3), 1167–1214 (2012)
3. Alvarez-Lacalle, E., Dorow, B., Eckmann, J.P., Moses, E.: Hierarchical structures induce long-range dynamical correlations in written texts. Proc. Natl. Acad. Sci. **103**(21), 7956–7961 (2006)
4. Ambikasaran, S., Foreman-Mackey, D., Greengard, L., Hogg, D.W., O'Neil, M.: Fast direct methods for Gaussian processes. IEEE Trans. Pattern Anal. Mach. Intell. **38**(2), 252–265 (2016). https://doi.org/10.1109/TPAMI.2015.2448083
5. Asmussen, S.: Applied Probability and Queues, 2nd edn. Springer (2003)

6. Bayer, C., Friz, P., Gatheral, J.: Pricing under rough volatility. Quant. Financ. **16**(6), 887–904 (2016)
7. Beran, J.: Statistical Methods for Data with Long-Range Dependence. Statistical Science, pp. 404–416 (1992)
8. Beran, J., Sherman, R., Taqqu, M.S., Willinger, W.: Long-range dependence in variable-bit-rate video traffic. IEEE Trans. Commun. **43**(2/3/4), 1566–1579 (1995)
9. Blanchet, J., Chen, X., Dong, J.: ε-Strong simulation for multidimensional stochastic differential equations via rough path analysis. Ann. Appl. Probab. **27**(1), 275–336 (2017)
10. Blanchet, J., Dong, J.: Perfect sampling for infinite server and loss systems. Adv. Appl. Probab. **47**(3), 761–786 (2015)
11. Blanchet, J., Dong, J., Liu, Z.: Exact sampling of the infinite horizon maximum of a random walk over a nonlinear boundary. J. Appl. Probab. **56**(1), 116–138 (2019)
12. Blanchet, J., Li, C.: Efficient simulation for the maximum of infinite horizon discrete-time Gaussian processes. J. Appl. Probab. **48**, 467–489 (2011)
13. Blanchet, J., Sigman, K.: On exact sampling of stochastic perpetuities. J. Appl. Probab. **48**(A), 165–182 (2011)
14. Blanchet, J., Wallwater, A.: Exact sampling of stationary and time-reversed queues. ACM Trans. Model. Comput. Simul. **25**(4), 26 (2015)
15. Bucklew, J.A., Radeke, R.: On the Monte Carlo simulation of digital communication systems in Gaussian noise. IEEE Trans. Commun. **51**(2), 267–274 (2003)
16. Chen, Y., Dong, J., Ni, H.: ε-strong simulation of fractional Brownian motion and related stochastic differential equations. Mathematics of Operations Research (2021)
17. Devroye, L.: Non-Uniform Random Variate Generation. Springer (1986)
18. Dieker, A.: Simulation of fractional Brownian motion. Ph.D. thesis, Masters Thesis, Department of Mathematical Sciences, University of Twente (2004)
19. Dieker, A.B., Mandjes, M.: On spectral simulation of fractional Brownian motion. Probab. Eng. Inf. Sci. **17**(3), 417–434 (2003)
20. Dieker, A.B., Mandjes, M.: Fast simulation of overflow probabilities in a queue with Gaussian input. ACM Trans. Model. Comput. Simul. **16**(2), 119–151 (2006)
21. Dietrich, C., Newsam, G.N.: Fast and exact simulation of stationary Gaussian processes through circulant embedding of the covariance matrix. SIAM J. Sci. Comput. **18**(4), 1088–1107 (1997)
22. Dombry, C., Engelke, S., Oesting, M.: Exact simulation of max-stable processes. Biometrika **103**(2), 303–317 (2016)
23. Doukhan, P., Oppenheim, G., Taqqu, M.: Theory and Applications of Long-Range Dependence. Springer Science & Business Media (2002)
24. Ensor, K., Glynn, P.: Simulating the maximum of a random walk. J. Stat. Plann. Inference **85**, 127–135 (2000)
25. Gatheral, J., Jaisson, T., Rosenbaum, M.: Volatility is rough. Quant. Financ. **18**(6), 933–949 (2018)
26. Glynn, P.W., Whitt, W.: The asymptotic efficiency of simulation estimators. Oper. Res. **40**(3), 505–520 (1992)
27. Heyde, C., Yang, Y.: On defining long-range dependence. J. Appl. Probab. **34**, 939–944 (1997)
28. Huang, C., Devetsikiotis, M., Lambadaris, I., Kaye, A.: Fast simulation of queues with long-range dependent traffic. Stoch. Model. **15**(3), 429–460 (1999)
29. Hurst, H.E.: Long-term storage capacity of reservoirs. Trans. Am. Soc. Civ. Eng. **116**(1), 770–799 (1951)
30. Jean-Francois, C.: Simulation and identification of the fractional Brownian motion: a bibliographical and comparative study. J. Stat. Softw. **5**, 1–53 (2000)
31. Karagiannis, T., Molle, M., Faloutsos, M.: Long-range dependence ten years of internet traffic modeling. IEEE Internet Comput. **8**(5), 57–64 (2004)
32. Lau, W.C., Erramilli, A., Wang, J.L., Willinger, W.: Self-similar traffic generation: The random midpoint displacement algorithm and its properties. In: Proceedings IEEE International Conference on Communications ICC'95, vol. 1, pp. 466–472. IEEE (1995)

33. Liu, Z., Blanchet, J., Dieker, A., Mikosch, T.: On logrithmically optimal exact simulation of max-stable and related random fields on a compact set. Bernoulli **25**(4A), 2949–2981 (2019)
34. Norros, I., Mannersalo, P., Wang, J.L.: Simulation of fractional Brownian motion with conditionalized random midpoint displacement. Adv. Perform. Anal. **2**(1), 77–101 (1999)
35. Robinson, P.M.: Gaussian Semiparametric Estimation of Long Range Dependence. The Annals of Statistics, pp. 1630–1661 (1995)
36. Robinson, P.M.: Time Series with Long Memory. Advanced Texts in Econometrics (2003)
37. Samorodnitsky, G.: Long Range Dependence. now Publishers Inc (2007)

Truncated Multivariate Student Computations via Exponential Tilting

Zdravko I. Botev and Yi-Lung Chen

Abstract In this paper we consider computations with the multivariate student density, truncated on a set described by a linear system of inequalities. Our goal is to both simulate from this truncated density, as well as to estimate its normalizing constant. To this end we consider an exponentially tilted sequential importance sampling (IS) density. We prove that the corresponding IS estimator of the normalizing constant, a rare-event probability, has bounded relative error under certain conditions. Along the way, we establish the multivariate extension of the Mill's ratio for the student distribution. We present applications of the proposed sampling and estimation algorithms in Bayesian inference. In particular, we construct efficient rejection samplers for the posterior densities of the Bayesian Constrained Linear Regression model, the Bayesian Tobit model, and the Bayesian smoothing spline for non-negative functions. Typically, sampling from such posterior densities is only viable via approximate Markov chain Monte Carlo (MCMC). Finally, we propose a novel *Reject-Regenerate* sampler, which is a hybrid between rejection sampling and MCMC. The Reject-Regenerate sampler creates a Markov chain, whose states are, with a certain probability, flagged as commencing a new regenerative or renewal cycle. Whenever a state initiates a new regenerative cycle, we can further flip a biased coin to decide whether the state is an exact draw from the target, or not. We show that the proposed MCMC algorithm is strongly efficient in a rare-event regime and provide a numerical example.

Keywords Truncated student · Genz estimator · Mill's ratio · Regenerative MCMC · Nummelin splitting

Z. I. Botev (✉) · Y.-L. Chen
School of Mathematics and Statistics, The University of New South Wales (UNSW Sydney), Sydney, NSW, Australia
e-mail: botev@unsw.edu.au

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_4

1 Introduction

A random vector $\boldsymbol{Y} \in \mathbb{R}^d$ is said to obey the standard multivariate student distribution, denoted by $\boldsymbol{Y} \sim \mathsf{t}_\nu$, if its density function is

$$c_1 \left(1 + \frac{1}{\nu}\|\boldsymbol{y}\|^2\right)^{-(\nu+d)/2}, \quad \boldsymbol{y} \in \mathbb{R}^d,$$

where $c_1 = \frac{\Gamma((d+\nu)/2)}{(\pi\nu)^{d/2}\Gamma(\nu/2)}$. Since the multivariate student is a location-scale family, if $\boldsymbol{Y} \sim \mathsf{t}_\nu$, we can write $\boldsymbol{\mu} + \Sigma^{1/2}\boldsymbol{Y} \sim \mathsf{t}_\nu(\boldsymbol{\mu}, \Sigma)$ for some location $\boldsymbol{\mu}$ and scale matrix Σ. For a given $m \times d$ real matrix C (we assume that $m \leq d$) and vectors $\boldsymbol{l}, \boldsymbol{u} \in \overline{\mathbb{R}}^m$ (here $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$), denoting $\ell = \mathbb{P}[\boldsymbol{l} \leq C\boldsymbol{Y} \leq \boldsymbol{u}]$, it follows that the density of $\boldsymbol{Y}$ conditioned on $\boldsymbol{Y} \in \{\boldsymbol{y} \,|\, \boldsymbol{l} \leq C\boldsymbol{y} \leq \boldsymbol{u}\}$ is

$$h(\boldsymbol{y}) = \frac{c_1 \left(1 + \frac{1}{\nu}\|\boldsymbol{y}\|^2\right)^{-(\nu+d)/2} \mathbb{1}\{\boldsymbol{l} \leq C\boldsymbol{y} \leq \boldsymbol{u}\}}{\ell}.$$

Estimating ℓ and simulating draws $\boldsymbol{Y} \sim h$ are two closely related problems with many statistical applications (see [10] and the references therein).

In this paper we consider estimating ℓ and simulating from h via the exponentially tilted sequential proposal density in [2], which is itself inspired by the separation-of-variables in [10] and constructs a proposal density g with following sequential form $(\boldsymbol{\theta} = (\theta_1, \ldots, \theta_d)^\top)$: $g(\boldsymbol{\theta}) = g(\theta_1)g_1(\theta_2 \,|\, \theta_1)g_2(\theta_3 \,|\, \theta_1, \theta_2) \ldots g_d(\theta_d \,|\, \theta_1, \ldots, \theta_{d-1})$, where each g_k belongs to a family of densities indexed by some 'tilting' parameter. An optimal tilting parameter is chosen such that the variance of an estimator of ℓ with IS density g is approximately minimized. This proposal density gives an accurate IS estimator for ℓ and an efficient rejection sampler to simulate draws from h (similar to ideas in [5]).

Our contributions over and above those in [2] are as follows. Firstly, we prove that the IS estimators for ℓ in [2] is asymptotically efficient. In particular, we show that the estimator enjoys the bounded relative error property for rare-event probability estimation. This is a desirable property in rare-event simulation [14]. A by-product of our theory is the extension of the well-known Mill's ratio [16] to the multivariate student distribution.

Secondly, we show how the exponentially tilted proposal density can be used in a rejection sampler for simulating draws from the posterior densities of the Bayesian constrained linear regression, Bayesian Tobit model, and the Bayesian smoothing spline for non-negative functions [6, 8, 9, 19]. When viable, this exact sampling is preferable to approximate MCMC posterior simulations.

Since exact sampling becomes inefficient as the dimensions grow, our third contribution is a novel *Reject-Regenerate* sampling algorithm. This sampling algorithm takes advantage of the classical splitting technique for Markov chains [18] and com-

bines regeneration with rejection sampling. We show that as an MCMC sampler in a rare-event regime, the algorithm is a strongly efficient MCMC, as defined in [4].

All truncated multivariate student code is freely available in both **R**[1] and **Matlab**.[2] There is also relevant JAVA software, see [3].

2 Review of the Sequentially Tilted Proposal Density

The construction of the proposal density in [2] begins by recalling that if $\boldsymbol{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$, and independently $R \sim \mathsf{chi}_\nu$, then $\sqrt{\nu}\boldsymbol{X}/R \sim \mathsf{t}_\nu(\mathbf{0}, \Sigma)$, where the density of a chi_ν random variable is

$$c_\nu(r) = \frac{2^{1-\nu/2}}{\Gamma(\nu/2)} \exp\left(-\frac{r^2}{2} + (\nu-1)\ln r\right), \quad r > 0.$$

In this manner, it suffices for one to consider simulating $(\boldsymbol{X}, R) \sim f$ where

$$f(\boldsymbol{x}, r) = \frac{\mathbb{1}\{r\boldsymbol{l} \le \sqrt{\nu}C\boldsymbol{x} \le r\boldsymbol{u}\}/\ell}{\sqrt{|\Sigma|}(2\pi)^{d/2} \times 2^{\nu/2-1}\Gamma(\nu/2)} \exp\left(-\frac{1}{2}\boldsymbol{x}^\top \Sigma^{-1}\boldsymbol{x} - \frac{r^2}{2} + (\nu-1)\ln r\right).$$

Next, let $\Sigma = L_1 L_1^\top$ be the Cholesky decomposition of Σ and $CL_1 = LQ$ be the LQ decomposition of CL_1 so that $L_1 \in \mathbb{R}^{d\times d}$, $L \in \mathbb{R}^{m\times d}$ are lower triangular, while $Q \in \mathbb{R}^{d\times d}$ is orthonormal. It follows that the substitution $\boldsymbol{x} = L_1 Q^\top \boldsymbol{z}$ yields the density

$$f(\boldsymbol{z}, r) = \frac{\mathbb{1}\{r\boldsymbol{l} \le \sqrt{\nu}L\boldsymbol{z} \le r\boldsymbol{u}\}/\ell}{\sqrt{|\Sigma|}(2\pi)^{d/2} \times 2^{\nu/2-1}\Gamma(\nu/2)} \exp\left(-\frac{\|\boldsymbol{z}\|^2}{2} - \frac{r^2}{2} + (\nu-1)\ln r\right). \tag{1}$$

Given $\boldsymbol{l}$, $\boldsymbol{u}$ and L, define $\mathfrak{R} = \{(\boldsymbol{z}, r) : r\boldsymbol{l} \le \sqrt{\nu}L\boldsymbol{z} \le r\boldsymbol{u}\}$. Then, we can write $\mathfrak{R}$ as

$$\begin{aligned}
&\tilde{l}_1(r) := \frac{r\,l_1}{\sqrt{\nu}}/L_{11} \le z_1 \le \frac{r\,u_1}{\sqrt{\nu}}/L_{11} =: \tilde{u}_1(r)\\
&\vdots\\
&\frac{\overbrace{\frac{r\,l_d}{\sqrt{\nu}} - \sum_{i=1}^{d-1} L_{di}z_i}^{\tilde{l}_d(r,z_1,\ldots,z_{d-1})}}{L_{dd}} \le z_d \le \frac{\overbrace{\frac{r\,u_d}{\sqrt{\nu}} - \sum_{i=1}^{d-1} L_{di}z_i}^{\tilde{u}_d(r,z_1,\ldots,z_{d-1})}}{L_{dd}}.
\end{aligned}$$

[1] https://cran.r-project.org/web/packages/TruncatedNormal/index.html.

[2] https://www.mathworks.com/matlabcentral/fileexchange/53796-truncated-normal-and-student-s-t-distribution-toolbox.

Observing that $\tilde{l}_k, \tilde{u}_k$ only depend on r and z_i for $i < k$, it is natural to consider a proposal g, with support on $\mathfrak{R}$, that takes sequential form in the following manner:

$$g(\boldsymbol{z}, r; \boldsymbol{\mu}, \eta) = g_0(r; \eta) g_1(z_1 \,|\, r; \mu_1) g_2(z_2 \,|\, r, z_1; \mu_2) \dots g_d(z_d \,|\, r, z_1, \dots, z_{d-1}; \mu_d),$$

where $\boldsymbol{\mu}$ and ν are parameters to be specified shortly. Denote $\phi(\cdot; \mu, \sigma^2)$ to be the pdf of the $\mathcal{N}(\mu, \sigma^2)$ distribution and Φ to be the cdf of $\mathcal{N}(0, 1)$. Then, one can choose $g_0(r; \eta) = \frac{\phi(r;\eta,1)}{\Phi(\eta)}$, $r > 0$ and

$$g_k(z_k \,|\, r, z_1, \dots, z_{k-1}; \mu_k) = \frac{\phi(z_k; \mu_k, 1) \mathbb{1}\{\tilde{l}_k \le z_k \le \tilde{u}_k\}}{\Phi(\tilde{u}_k - \mu_k) - \Phi(\tilde{l}_k - \mu_k)}, \; k = 1, 2, \dots,$$

that is (denoting $\mathcal{TN}_{[a,b]}(\theta, \sigma^2)$ as a $\mathcal{N}(\theta, \sigma^2)$ random variable, truncated/conditioned to the interval $[a, b]$):

$$\begin{aligned} R &\sim \mathcal{TN}_{(0,\infty)}(\eta, 1) \\ Z_k \,|\, R, Z_1, \dots, Z_{k-1} &\sim \mathcal{TN}_{(\tilde{l}_k, \tilde{u}_k)}(\mu_k, 1), \quad k = 1, \dots, d. \end{aligned} \tag{2}$$

Finally, define the logarithm of the likelihood ratio:

$$\begin{aligned} \psi(\boldsymbol{z}, r; \boldsymbol{\mu}, \eta) &:= \ln \frac{c_\nu(r) \phi_{I_d}(\boldsymbol{z})}{g(\boldsymbol{z}, r; \boldsymbol{\mu}, \eta)} \\ &= \frac{\|\boldsymbol{\mu}\|^2}{2} - \boldsymbol{z}^\top \boldsymbol{\mu} + \frac{\eta^2}{2} - r\eta + (\nu - 1) \ln r + \ln \Phi(\eta) \\ &\quad + \sum_{k=1}^{d} \ln[\Phi(\tilde{u}_k - \mu_k) - \Phi(\tilde{l}_k - \mu_k)]. \end{aligned}$$

The tilting parameters $(\eta, \boldsymbol{\mu})$ is the unique solution to the program [2]

$$(\boldsymbol{z}^*, r^*, \boldsymbol{\mu}^*, \eta^*) = \operatorname*{argmin}_{(\boldsymbol{\mu}, \eta)} \operatorname*{argmax}_{(\boldsymbol{z}, r) \in \mathfrak{R}} \psi(\boldsymbol{z}, r; \boldsymbol{\mu}, \eta). \tag{3}$$

Intuitively, $\max_{\boldsymbol{z}, r \in \mathfrak{R}} \psi$ seeks to find how much g can deviate from f. The optimization with respect to $(\boldsymbol{\mu}, \eta)$ shapes g so that this worst-case deviation is minimized. This optimization is tackled in [2] by solving the nonlinear system of equations $\nabla \psi = \mathbf{0}$, where $\nabla \psi$ is the gradient with respect to all the variables. To this end, [2] proposes the unbiased IS estimator

$$\hat{\ell} = \frac{1}{n} \sum_{k=1}^{n} \exp(\psi(\boldsymbol{Z}_k, R_k; \boldsymbol{\mu}^*, \eta^*)), \quad (\boldsymbol{Z}_k, R_k) \sim_{iid} g. \tag{4}$$

Since $\psi(z, r; \boldsymbol{\mu}^*, \eta^*) \leq c =: \psi(z^*, r^*, \boldsymbol{\mu}^*, \eta^*)$ for all (z, r), the following rejection sampling algorithm yields an exact draw $(\boldsymbol{Z}, R) \sim f$.

Algorithm 1: Rejection sampling for f in (1)

Input: $(z^*, r^*, \boldsymbol{\mu}^*, \eta^*)$, the solution to program (3) and $c \leftarrow \psi(z^*, r^*, \boldsymbol{\mu}^*, \eta^*)$

repeat
  Draw $(\boldsymbol{Z}, R) \sim g(\cdot, \cdot; \boldsymbol{\mu}^*, \eta^*)$, as given by (2);
  Independently draw $E \sim \mathsf{Exp}(1)$;
until $E \geq c - \psi(\boldsymbol{Z}, R; \boldsymbol{\mu}^*, \eta^*)$;
return $(\boldsymbol{Z}, R)$ *an exact draw from* f

3 Asymptotic Efficiency of the IS Estimator

The main contribution presented in this section is Theorem 3, which establishes an asymptotic efficiency of the estimator (4). Along the way, we shall establish Theorem 2, which is a multivariate extension of the Mill's ratio for the student distribution. We shall begin by establishing the following notations.

Suppose Σ is a positive definite covariance matrix and $\nu > 0$ is degrees of freedom. We wish to find the asymptotic approximation to the tail

$$\ell(\gamma) = \mathbb{P}[\boldsymbol{Y} \geq \boldsymbol{l}(\gamma)], \quad \boldsymbol{Y} \sim \mathsf{t}_\nu(\boldsymbol{0}, \Sigma) \tag{5}$$

where $\max_i l_i > 0$, and at least one component of $\boldsymbol{l}(\gamma)$ diverges to ∞, that is, $\lim_{\gamma \uparrow \infty} \|\boldsymbol{l}(\gamma)\| = \infty$. Let P be a permutation matrix which maps the vector $(1, \ldots, d)^\top$ into the permutation $\boldsymbol{p} = (p_1, \ldots, p_d)^\top$, that is, $P(1, \ldots, d)^\top = \boldsymbol{p}$. Note that $\ell(\gamma) = \mathbb{P}[P\boldsymbol{Y} \geq P\boldsymbol{l}(\gamma)]$ for any permutation $\boldsymbol{p}$, and $P\boldsymbol{Y} \sim \mathsf{t}_\nu(\boldsymbol{0}, P\Sigma P^\top)$. We will specify $\boldsymbol{p}$ shortly.

Consider the convex quadratic programming: $\min_{\boldsymbol{x}: \boldsymbol{x} \geq P\boldsymbol{l}(\gamma)} \boldsymbol{x}^\top (P\Sigma P^\top)^{-1} \boldsymbol{x}$. The Karush-Kuhn-Tucker (KKT) conditions [12, p. 409] are a necessary and sufficient condition to find the unique solution:

$$\begin{aligned} (P\Sigma P^\top)^{-1}\boldsymbol{x} - \boldsymbol{\lambda} &= \boldsymbol{0} \\ \boldsymbol{\lambda} \geq \boldsymbol{0}, \quad P\boldsymbol{l} - \boldsymbol{x} &\leq \boldsymbol{0} \\ \boldsymbol{\lambda}^\top (P\boldsymbol{l} - \boldsymbol{x}) &= 0\,, \end{aligned} \tag{6}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^d$ is the Lagrange multiplier. Denote the number of active constraints in the quadratic program by d_1 and the number of inactive constraints as d_2, so that $d_1 + d_2 = d$. Note that the number of active constraints $d_1 \geq 1$, because otherwise the solution is $\boldsymbol{x} = \boldsymbol{0}$, which implies $P\boldsymbol{l} \leq \boldsymbol{0}$, thus reaching a contradiction.

Given the partition $\boldsymbol{\lambda} = (\boldsymbol{\lambda}_1^\top, \boldsymbol{\lambda}_2^\top)^\top$ with $\dim(\boldsymbol{\lambda}_1) = d_1$ and $\dim(\boldsymbol{\lambda}_2) = d_2$, one can select the permutation vector $\boldsymbol{p}$ and the corresponding matrix P in such a way that all

the active constraints in (6) correspond to $\boldsymbol{\lambda}_1 > \mathbf{0}$ and all the inactive ones to $\boldsymbol{\lambda}_2 = \mathbf{0}$. For simplicity of the notation, we assume that this reordering of the variables via the permutation operator P is always applied to $\boldsymbol{l}$ and Σ, so that $P\boldsymbol{l} = \boldsymbol{l}$ and $P\Sigma P^\top = \Sigma$. If we partition $\boldsymbol{x}, \boldsymbol{l}$, and $\Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}$, then the KKT equations tell us that the optimal solution $\boldsymbol{x}^*$ is:

$$\begin{aligned} \boldsymbol{x}_1^* &= \Sigma_{11}\boldsymbol{\lambda}_1 = \boldsymbol{l}_1(\gamma) \\ \boldsymbol{x}_2^* &= \Sigma_{21}\boldsymbol{\lambda}_1 = \Sigma_{21}\Sigma_{11}^{-1}\boldsymbol{l}_1(\gamma) > \boldsymbol{l}_2(\gamma) \end{aligned}$$

with the global minimum $\frac{1}{2}(\boldsymbol{x}^*)^\top \Sigma^{-1}\boldsymbol{x}^* = \frac{1}{2}(\boldsymbol{x}_1^*)^\top \boldsymbol{\lambda}_1 = \frac{1}{2}\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1$.

Theorem 1 (Mill's Ratio For Multivariate Normal [11]) *Under the conditions above, if $\boldsymbol{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$, then as $\gamma \uparrow \infty$, we have:*

$$\mathbb{P}[\boldsymbol{X} \geq \boldsymbol{l}(\gamma)] = \frac{\mathbb{P}[\boldsymbol{X}_2 \geq \boldsymbol{l}_\infty \mid \boldsymbol{X}_1 = \mathbf{0}]}{(2\pi)^{d_1/2}|\Sigma_{11}|^{1/2}\prod_{k=1}^{d_1} \boldsymbol{u}_k^\top \Sigma_{11}^{-1}\boldsymbol{l}_1} \exp\left(-\frac{\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{2}\right)(1 + o(1)),$$

where $\boldsymbol{l}_\infty := \lim_{\gamma\uparrow\infty}(\boldsymbol{l}_2(\gamma) - \boldsymbol{x}_2^*(\gamma))$ with $\boldsymbol{l}_\infty \leq \mathbf{0}$.

One of our main contributions is to generalize the result of [11] to the following. The proof for this result is provided in the Appendix.

Theorem 2 (Mill's Ratio For Multivariate Student) *Suppose $\boldsymbol{Y} \sim \mathsf{t}_\nu(\mathbf{0}, \Sigma)$ with $\nu > 0$, and Σ and $\boldsymbol{l}$ satisfy the conditions imposed for the solution of (6). Then,*

$$\mathbb{P}[\boldsymbol{Y} \geq \boldsymbol{l}(\gamma)] = (c + o(1)) \times \left(1 + \frac{\boldsymbol{l}_1(\gamma)^\top \Sigma_{11}^{-1}\boldsymbol{l}_1(\gamma)}{\nu}\right)^{-\nu/2}, \qquad \gamma \uparrow \infty,$$

where c is a constant, independent of γ, and is given by the expression:

$$c = \frac{2^{1-\nu/2}}{\Gamma(\nu/2)} \int_0^\infty r^{\nu-1}\mathbb{P}[\boldsymbol{X} \geq r\boldsymbol{l}_\infty]\mathrm{d}r,$$

with $\boldsymbol{l}_\infty = \lim_{\gamma\uparrow\infty} \frac{\boldsymbol{l}(\gamma)}{\sqrt{\nu + \boldsymbol{l}_1(\gamma)^\top \Sigma_{11}^{-1}\boldsymbol{l}_1(\gamma)}}$ and $\boldsymbol{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$.

As an example, when $d = 1$ and $l^* = 1$, $L_1 = \sigma$, we obtain:

$$\ell(\gamma) \downarrow \frac{2^{1-\nu/2}}{\Gamma(\frac{\nu}{2})\left(1 + \frac{\gamma^2\sigma^2}{\nu}\right)^{\nu/2}} \int_0^\infty u^{\nu-1}\overline{\Phi}(u)\mathrm{d}u = \frac{\Gamma((\nu+1)/2)}{\nu\sqrt{\pi}\Gamma(\frac{\nu}{2})\left(1 + \frac{\gamma^2\sigma^2}{\nu}\right)^{\nu/2}}.$$

The last agrees with the result of [20, 21], namely, $\ell(\gamma) \downarrow \frac{\gamma}{\sigma\nu} t_\nu(\gamma; 0, \sigma^2)$, where $t_\nu(x; \mu, \sigma^2)$ is the pdf of the univariate $t_\nu(\mu, \sigma^2)$ distribution evaluated at x.

A second example considers the tail asymptotics of $\boldsymbol{l}(\gamma) = \gamma \boldsymbol{l}$, $\max_i l_i > 0$. We have:

$$\mathbb{P}[\boldsymbol{Y} \geq \gamma \boldsymbol{l}] = (c + o(1)) \times \left(1 + \gamma^2 \frac{\boldsymbol{l}_1^\top \Sigma_{11}^{-1} \boldsymbol{l}_1}{\nu}\right)^{-\nu/2}, \qquad \gamma \uparrow \infty,$$

where $c = \frac{2^{1-\nu/2}}{\Gamma(\nu/2)} \int_0^\infty r^{\nu-1} \mathbb{P}\left[\boldsymbol{X} \geq \frac{r\boldsymbol{l}}{\sqrt{\boldsymbol{l}_1^\top \Sigma_{11}^{-1} \boldsymbol{l}_1}}\right] \mathrm{d}r$.

Finally, recall that $g(x) = \Theta(f(x))$ is the same as $g(x) = \mathcal{O}(f(x))$ and $f(x) = \mathcal{O}(g(x))$. In addition to $f(x) = o(g(x))$ being a shorthand notation for $\lim_{x\uparrow\infty} f(x)/g(x) = 0$, we use the notation $\lesssim$ for "asymptotically less than". We have the following result concerning the asymptotic efficiency of this IS estimator for ℓ. The proof is provided in the Appendix.

Theorem 3 (Bounded Relative Error estimator) *Suppose we wish to estimate the tail probability* $\ell(\gamma) = \mathbb{P}[\boldsymbol{Y} \geq \boldsymbol{l}(\gamma)]$, *where* $\boldsymbol{Y} \sim \mathsf{t}_\nu(\boldsymbol{0}, \Sigma)$, *and* $\max_i l_i > 0$ *with* $\boldsymbol{l}(\gamma)/\gamma = \Theta(\boldsymbol{1})$ *as* $\gamma \uparrow \infty$. *Then, the exponentially tilted estimator*

$$\hat{\ell} = \exp(\psi(\boldsymbol{Z}, R; \boldsymbol{\mu}^*, \eta^*)), \qquad (\boldsymbol{Z}, R) \sim g(\boldsymbol{z}, r; \eta^*, \boldsymbol{\mu}^*),$$

is a bounded relative error estimator: lim sup$_{\gamma\uparrow\infty} \frac{\mathrm{Var}(\hat{\ell})}{\ell^2(\gamma)} < \infty$.

We note that the same estimator $\hat{\ell}$ in Theorem 3 was shown to enjoy a vanishing relative error property, $\frac{\mathrm{Var}(\hat{\ell})}{\ell^2(\gamma)} \downarrow 0$, as $\nu \uparrow \infty$, see [1].

Theorem 3 explains the excellent simulation results obtained in [2], especially when estimating small tail probabilities. In view of these theoretical and simulation results, we can confidently state that the method proposed in [2] significantly outperforms its competitors, such as Genz's method [10]. Figure 1 summarizes one example of estimating $\mathbb{P}(\boldsymbol{0} \leq C\boldsymbol{Y} \leq \boldsymbol{2})$ with $\boldsymbol{Y} \sim \mathsf{t}_{10}(\boldsymbol{0}, I_d)$ and $(CC^\top)^{-1} = \frac{1}{2} I_d + \frac{1}{2} \boldsymbol{1}\boldsymbol{1}^\top$, where $\boldsymbol{1} = (1, \ldots, 1)^\top \in \mathbb{R}^d$.

d	$\hat{\ell}_{\text{Genz}}$	$\hat{\ell}$ (relative error)
5	0.001 723 178 855 987 53	0.001 724 959 (0.002 808 291)
10	$1.419\,398\,621\,879\,78 \times 10^{-7}$	$1.459\,521 \times 10^{-7}$ ($2.545\,927 \times 10^{-7}$)
20	$5.438\,369\,348\,166\,98 \times 10^{-17}$	$2.851\,725 \times 10^{-17}$ ($5.249\,063 \times 10^{-17}$)
30	$4.217\,329\,746\,075\,88 \times 10^{-31}$	$3.687\,894 \times 10^{-28}$ ($6.943\,565 \times 10^{-28}$
40	$1.004\,799\,835\,397\,11 \times 10^{-44}$	$8.325\,791 \times 10^{-40}$ ($1.584\,754 \times 10^{-39}$)
50	$9.065\,508\,271\,833\,77 \times 10^{-60}$	$5.166\,692 \times 10^{-52}$ ($9.901\,967 \times 10^{-52}$)
100	0	$1.711\,223 \times 10^{-118}$ ($3.334\,542 \times 10^{-118}$)
150	0	$1.030\,134 \times 10^{-190}$ ($2.028\,947 \times 10^{-190}$)

Fig. 1 Comparison of proposed estimator with that of Genz [10]. Both the computing time and the relative error favor $\hat{\ell}$. In fact, Genz's estimator gives meaningful estimates only for up to $d = 10$

4 Application to Constrained Linear Regression

Consider the linear regression model $\boldsymbol{Y} = X\boldsymbol{\beta} + \boldsymbol{\epsilon}$, $X \in \mathbb{R}^{m\times d}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2 I)$ with the (possibly "improper" Bayesian) prior information $p(\boldsymbol{\beta}) \propto \mathbb{1}\{\boldsymbol{l} \le C\boldsymbol{\beta} \le \boldsymbol{u}\}$ for some appropriate matrix C and vectors $\boldsymbol{l}, \boldsymbol{u}$.

Assuming for simplicity a non-informative prior $p(\sigma) \propto \sigma^{-2}$, the Bayesian posterior from which we wish to sample is: $f(\boldsymbol{\beta}, \sigma) \propto \exp\left(-\frac{\|\boldsymbol{y}-X\boldsymbol{\beta}\|^2}{2\sigma^2} - (m+2)\ln\sigma\right) \times \mathbb{1}\{\boldsymbol{l} \le C\boldsymbol{\beta} \le \boldsymbol{u}\}$. If $\hat{\boldsymbol{\beta}}$ is the least squares estimate, and $s^2 := \|\boldsymbol{y} - X\hat{\boldsymbol{\beta}}\|^2$ is the norm of the residuals, then

$$f(\boldsymbol{\beta}, \sigma) \propto \exp\left(-\frac{s^2}{2\sigma^2} - (m+2)\ln\sigma - \frac{(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}})^\top X^\top X(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}})}{2\sigma^2}\right) \times \mathbb{1}\{\boldsymbol{l} \le C\boldsymbol{\beta} \le \boldsymbol{u}\}$$

Let $L_1 L_1^\top = X^\top X$ be the lower triangular Cholesky decomposition of $X^\top X$ and $LQ = CL_1^{-\top}$ be the LQ decomposition of matrix $CL_1^{-\top}$. Then, the bijective smooth transformation

$$\begin{aligned} &r = s/\sigma, \quad \boldsymbol{z} = Q\,L_1^\top(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}})/\sigma \\ &\boldsymbol{l} \leftarrow \sqrt{\nu}(\boldsymbol{l} - C\hat{\boldsymbol{\beta}})/s, \qquad \boldsymbol{u} \leftarrow \sqrt{\nu}(\boldsymbol{u} - C\hat{\boldsymbol{\beta}})/s, \end{aligned} \tag{7}$$

where $\nu \leftarrow (m-d+1) \ge 1$, yields the density for $(\boldsymbol{z}, r)$:

$$f(\boldsymbol{z}, r) \propto \exp\left(-\frac{1}{2}\|\boldsymbol{z}\|^2 - \frac{r^2}{2} + (\nu-1)\ln r\right) \times \mathbb{1}\{r\boldsymbol{l} \le \sqrt{\nu}L\boldsymbol{z} \le r\boldsymbol{u}\},$$

which is finally of a form amenable to Algorithm 1.

As a numerical example, we consider the `Apple dataset` [7] which records 207 observations of the number apples produced (in cartons) along with the number of trees of each year of age from various growers. This can be modeled by the Bayesian constrained linear regression where the i-th response, $y_i \in \mathbb{R}$, is the number of apples produced and the i-th predictor vector, $\boldsymbol{x}_i \in \mathbb{R}^{10}$, records the number of trees of age $j+1$, $j = 1, \ldots, 10$ being the entry index within the vector $\boldsymbol{x}_i$. Note that here trees of year 1 is considered to have zero production and `age 11` is considered as the mature age of an apple tree, so that any tree above an age of 11 is recorded as `age 11`. Finally, the prior $\pi(\boldsymbol{\beta}) \propto \mathbb{1}\{\beta_1 \le \beta_2 \le \cdots \le \beta_{10}\}$ captures the prior belief that a more mature tree produces more apples. The results are summarized in Fig. 2 and Table 1.

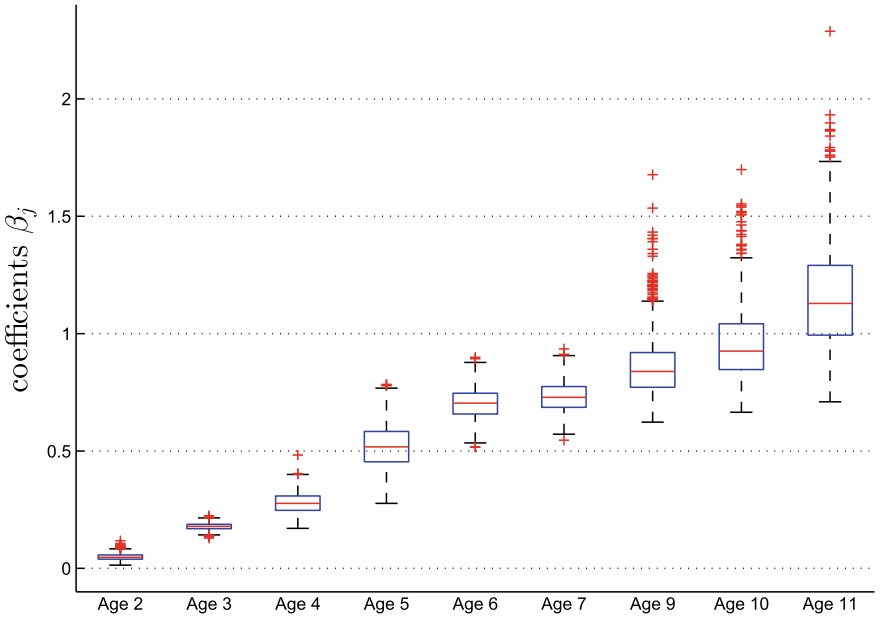


Fig. 2 The empirical posterior distribution derived from $n = 10^4$ independent exact draws

Table 1 Estimated mean, 0.95 credible interval and standard deviation

	Mean	0.025-quantile	0.975-quantile	Sample std.
Age 2	3.1890×10^{-2}	6.1072×10^{-3}	5.4554×10^{-2}	1.1964×10^{-2}
Age 3	4.8505×10^{-2}	2.5604×10^{-2}	7.7669×10^{-2}	1.3294×10^{-2}
Age 4	1.7888×10^{-1}	1.5264×10^{-1}	2.0631×10^{-1}	1.3757×10^{-2}
Age 5	2.7876×10^{-1}	2.0196×10^{-1}	3.6938×10^{-1}	4.3667×10^{-2}
Age 6	5.2097×10^{-1}	3.5380×10^{-1}	7.0809×10^{-1}	9.1928×10^{-2}
Age 7	7.0249×10^{-1}	5.8087×10^{-1}	8.2818×10^{-1}	6.4465×10^{-2}
Age 8	7.3122×10^{-1}	6.1550×10^{-1}	8.5433×10^{-1}	6.2582×10^{-2}
Age 9	8.6244×10^{-1}	6.8128×10^{-1}	1.2004	1.3217×10^{-1}
Age 10	9.5653×10^{-1}	7.2493×10^{-1}	1.3117	1.5596×10^{-1}
Age 11	1.1594	8.2762×10^{-1}	1.6596	2.1844×10^{-1}

5 Tobit Model Application

In the Tobit regression model with normally distributed error, the response is $Y_i = \max\{W_i, u_i\}, \forall i$, where $\boldsymbol{W} \sim \mathcal{N}(X\boldsymbol{\beta}, \sigma^2 I)$. The posterior, given for the data $\boldsymbol{y}$ and with uninformative priors, say $p(\boldsymbol{\beta}) \propto 1$ and $p(\sigma) \propto \sigma^{-2}$, is then of the form:

$$f(\boldsymbol{\beta},\sigma) \propto \exp\left(-\sum_{i:y_i>u_i}\left(\frac{(y_i-\boldsymbol{x}_i^\top\boldsymbol{\beta})^2}{2\sigma^2}+\ln\sigma\right)+\sum_{i:y_i=u_i}\ln\Phi((u_i-\boldsymbol{x}_i^\top\boldsymbol{\beta})/\sigma)\right)\times\sigma^{-2}$$

Let $\overline{\boldsymbol{y}}$ and $\underline{\boldsymbol{y}}$ be vectors that collect all $y_i > u_i$ and $y_i = u_i$, respectively. Denote the corresponding matrix with predictors via $\overline{X}$ and $\underline{X}$, respectively. Using a latent variable w_i for each $y_i = u_i$, we can write:

$$f(\boldsymbol{\beta},\sigma,\boldsymbol{w}) \propto \exp\left(-\frac{\|\overline{\boldsymbol{y}}-\overline{X}\boldsymbol{\beta}\|^2}{2\sigma^2}-\frac{\|\boldsymbol{w}-\underline{X}\boldsymbol{\beta}\|^2}{2\sigma^2}-(m+2)\ln\sigma\right)\mathbb{1}\{\boldsymbol{w}\le\boldsymbol{u}\}$$

so that the marginal of $(\boldsymbol{\beta},\sigma)$ has the desired posterior pdf. Note that, conditional on $(\sigma,\boldsymbol{w})$, the distribution of $\boldsymbol{\beta}$ is $\mathcal{N}(C(\overline{X}^\top\overline{\boldsymbol{y}}+\underline{X}^\top\boldsymbol{w}),\sigma^2C)$, where $C^{-1}=\overline{X}^\top\overline{X}+\underline{X}^\top\underline{X}$. Thus, to simulate from the posterior, it suffices to simulate from the marginal of $(\sigma,\boldsymbol{w})$, which is of the form:

$$f(\sigma,\boldsymbol{w}) \propto \exp\left(-\frac{\|\boldsymbol{w}\|^2}{2\sigma^2}+\frac{(\overline{X}^\top\overline{\boldsymbol{y}}+\underline{X}^\top\boldsymbol{w})^\top C(\overline{X}^\top\overline{\boldsymbol{y}}+\underline{X}^\top\boldsymbol{w})}{2\sigma^2}-\frac{\|\overline{\boldsymbol{y}}\|^2}{2\sigma^2}\right)\mathbb{1}\{\boldsymbol{w}\le\boldsymbol{u}\}\times\sigma^{d-m-2}.$$

After some straightforward computations we can rewrite it as:

$$f(\sigma,\boldsymbol{w}) \propto \exp\left(-\frac{(\boldsymbol{w}-\hat{\boldsymbol{w}})^\top(I-\underline{X}C\underline{X}^\top)(\boldsymbol{w}-\hat{\boldsymbol{w}})}{2\sigma^2}-\frac{s^2}{2\sigma^2}-(m-d+2)\ln\sigma\right)\mathbb{1}\{\boldsymbol{w}\le\boldsymbol{u}\},$$

where $\hat{\boldsymbol{w}} := \underline{X}(\overline{X}^\top\overline{X})^{-1}\overline{X}^\top\overline{\boldsymbol{y}}$ and $s^2 := \overline{\boldsymbol{y}}^\top(I-\overline{X}(\overline{X}^\top\overline{X})^{-1}\overline{X}^\top)\overline{\boldsymbol{y}}$. It follows that the transformation $r=s/\sigma$, $\boldsymbol{z}=L^{-1}(\hat{\boldsymbol{w}}-\boldsymbol{w})/\sigma$, where $LL^\top = I+\underline{X}(\overline{X}^\top\overline{X})^{-1}\underline{X}^\top$ is the Cholesky decomposition, and $\nu \leftarrow m-d-\dim(\underline{\boldsymbol{y}})+1$, $\boldsymbol{l}\leftarrow\sqrt{\nu}(\hat{\boldsymbol{w}}-\boldsymbol{u})/s$, reveals that simulating from $f(\sigma,\boldsymbol{w})$ is equivalent to simulating form

$$f(\boldsymbol{z},r) \propto \exp\left(-\frac{\|\boldsymbol{z}\|^2}{2}-\frac{r^2}{2}+(\nu-1)\ln r\right)\mathbb{1}\{\sqrt{\nu}\,L\boldsymbol{z}\ge r\boldsymbol{l}\},$$

which is again amenable to Eq. (1). As a numerical example, we consider the Women's Wages dataset [17] with $u_i = 0\forall i$. It consists of $m = 753$ observations on the number of hours per annum (the response y_i) married women spend in the labor force. The seven predictor variables $(x_1,\ldots,x_7)$ are:

1. `kidslt6`, number of children of age less than 6;
2. `kidsge6`, number of children of age between 6 and 18;
3. `age`, age of the married woman;
4. `educ`, level of education;
5. `experience`, number of years worked since age 18;
6. `nwifeinc`, household income that is not earned by the wife;

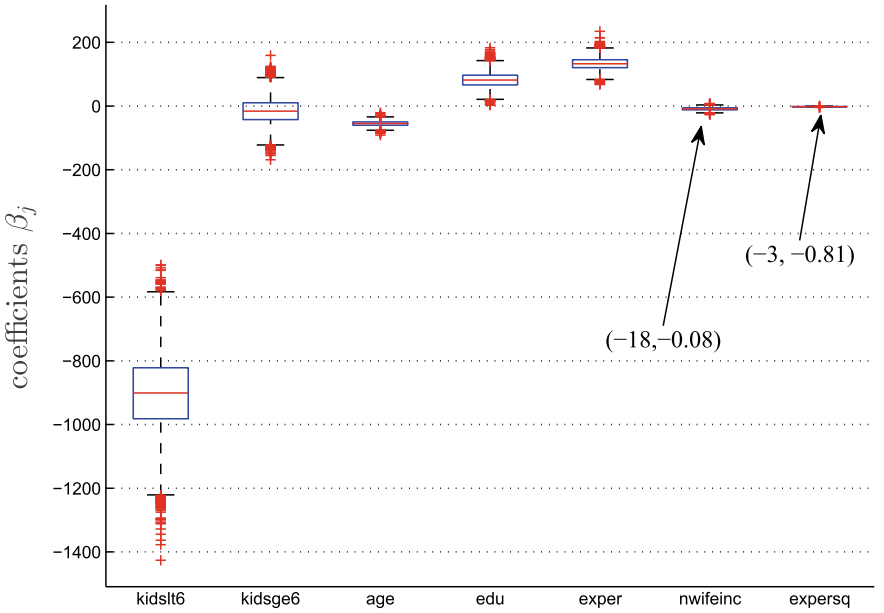


Fig. 3 The empirical posterior distribution (represented a boxplots) derived from $n = 10^4$ independent exact draws

Table 2 Estimated mean, 0.95 Bayesian credible interval and standard deviation of the posterior

	Mean	0.025-quantile	0.975-quantile	Sample std.
β_0	9.5938×10^2	2.2870×10^1	1.8375×10^3	4.6362×10^2
kidslt6	-9.0330×10^2	-1.1493×10^3	-6.8019×10^2	1.1975×10^2
kidsge6	-1.6187×10^1	-9.3890×10^1	6.1027×10^1	3.9863×10^1
age	-5.5046×10^1	-7.0864×10^1	-4.0027×10^1	7.8629
educ	8.1827×10^1	3.8144×10^1	1.2759×10^2	2.2705×10^1
exper	1.3301×10^2	9.7990×10^1	1.7019×10^2	1.8504×10^1
nwifeinc	−8.9230	-1.8092×10^1	8.0835×10^{-2}	4.6469
expersq	−1.8884	−3.0007	-8.0634×10^{-1}	5.5906×10^{-1}

7. `expersq`, square of the number of years the married woman has worked since age 18.

The results are summarized in Fig. 3 and Table 2.

We can see that the most important factors for women's labour force participation is: (1) the number of children of age less than 6 (with a large negative effect on the number of hours in the workforce); (2) the experience in the work force (with a large positive effect). Education and Age are also relevant, but their effect is smaller (the corresponding coefficient estimates are much smaller).

6 Application to "Bayesian" Splines for Non-negative Functions

Consider the dataset $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$. Using $\{0, x_1, \ldots, x_n, h\}$ as knots, a common cubic smoothing spline regression model is: $y_i = \sum_{k=1}^{n+4} \beta_k s_k(x_i) + \epsilon_i$, $\epsilon_i \sim_{iid} \mathcal{N}(0, \sigma^2)$, where s_k is the k-th 4-th order B-spline basis for inner knots $\{x_1, \ldots, x_n\}$. The goal is to estimate $\boldsymbol{\beta} := (\beta_1, \ldots, \beta_{n+4})^\top$ such that the estimator is non-negative [19]. Without the non-negativity constraints, the frequentist estimator is:

$$\underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} \left(y_i - \sum_{k=1}^{n+4} \beta_k s_k(x_i) \right)^2 + \lambda \int_0^h \left(\sum_{k=1}^{n+4} \beta_k s_k''(x) \right)^2 \mathrm{d}x,$$

where $\lambda > 0$ controls the smoothness of the splines. Denoting

$$\boldsymbol{s}(x_i) = (s_1(x_i), s_2(x_i), \ldots, s_{n+4}(x_i))^\top,$$

the Bayesian inference proceeds with [19]:

$$f(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\beta}, \sigma^2) \propto \sigma^{-n} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \boldsymbol{\beta}^\top \boldsymbol{s}(x_i))^2 \right)$$
$$f(\boldsymbol{\beta}|\sigma^2, \lambda) \propto \lambda^{(n+4)/2} \sigma^{-(n+4)} \exp\left(-\frac{\lambda}{2\sigma^2} \boldsymbol{\beta}^\top K \boldsymbol{\beta} \right), \quad p(\sigma^2) \propto \sigma^{-2},$$

where K is a square matrix of size $n+4$ with entries $K_{kl} = \int_0^h s_k''(x) s_l''(x)\, \mathrm{d}x$. Enforcing the non-negativity of the regression function over the grid $\{0 \le z_1 < z_2 < \cdots < z_m \le h\}$ reduces to imposing the constraint: $\boldsymbol{\beta}^\top \boldsymbol{s}(z_j) := \sum_{k=1}^{n+4} \beta_k s_k(z_j) \ge 0$, $j = 1, \ldots, m$. Consequently, Bayesian inference for this model requires one to sample from the posterior distribution:

$$f(\boldsymbol{\beta}, \sigma^2|\boldsymbol{y}, \boldsymbol{x}, \lambda) \propto f(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\beta}, \sigma^2) f(\boldsymbol{\beta}|\sigma^2, \lambda\pi(\sigma^2),$$

restricted to: $\boldsymbol{\beta}^\top \boldsymbol{s}(z_j) \ge 0$, $j = 1, \ldots, m$. By denoting $S = [\boldsymbol{s}(x_1), \ldots, \boldsymbol{s}(x_n)]^\top$ and completing the square, the posterior distribution reduces to:

$$f(\boldsymbol{\beta}, \sigma^2|\boldsymbol{y}, \boldsymbol{x}, \lambda) \propto \exp\left(-\frac{s^2}{2\sigma^2} - (2n+6) \ln \sigma - \frac{(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})^\top A (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})}{2\sigma^2} \right) \mathbb{1}\{S\boldsymbol{b} \ge \boldsymbol{0}\},$$

where $A = S^\top S + \lambda K$, $\hat{\boldsymbol{\beta}} = A^{-1} S^\top \boldsymbol{y}$, $s^2 = \boldsymbol{y}^\top \boldsymbol{y} - \boldsymbol{y}^\top S A^{-1} S \boldsymbol{y}$. This again takes the form amenable to Algorithm 1. Figure 4 gives a numerical example from [19].

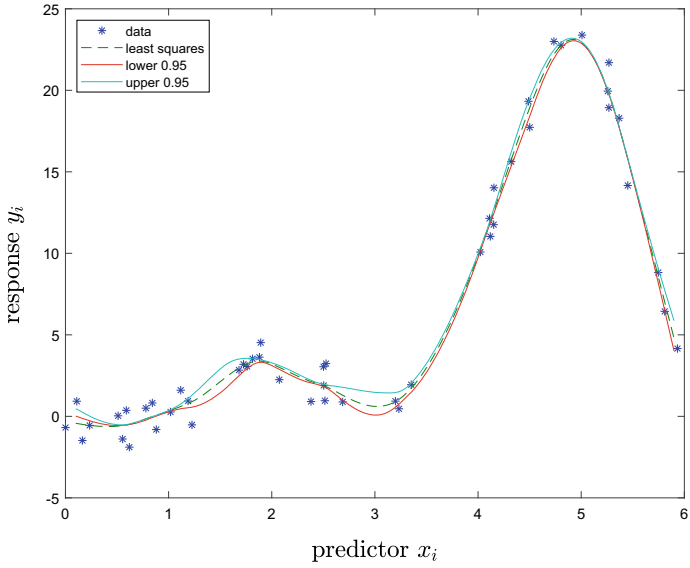


Fig. 4 There are fifty x_i's uniformly distributed on $[0, 2\pi]$ and $y_i = x_i \sin^2(x_i) + \epsilon_i$, where $\epsilon_i \sim \mathsf{N}(0, 1)$. The dotted line and the bands are the mean and the empirical 95% function values obtained from sampling the posterior distribution exactly 1000 times

7 The Reject-Regenerate Sampler

So far we have successfully applied the exponential tilting technique in [2] to construct exact samplers for certain Bayesian posterior densities. However, due to the curse of dimensionality, no matter how careful one constructs a proposal density, rejection sampling will eventually become inefficient as the sampling dimension grows.

Consider the situation where we have designed a sequential proposal density for efficient rejection sampling. We know that the rejection sampler will be efficient up to a certain dimension, which is typically unknown a-priori. Beyond this unknown dimension, the rejection sampling will be inefficient and we will have to switch from exact rejection sampling to approximate independent Metropolis sampling [13]. This scenario has a number of undesirable features.

First, the user has to explicitly decide when a rejection sampler is inefficient. For example, should the cutoff for efficiency be an acceptance probability of 10^{-3} or 10^{-2}?

Second, the user has to run the rejection sampling algorithm to find out if it meets the efficiency criterion above. In the likely event that the rejection sampler does not meet the efficient criterion, this simulation effort has been effectively wasted, because the user now has to run a separate MCMC algorithm from scratch. The simulation effort from rejection sampling is not recycled by the MCMC sampler, but is simply used to make a dichotomous, all-or-nothing decision about the rejection sampler.

Given the above drawbacks of using rejection and MCMC sampling as two distinct algorithms, in this section we propose a single algorithm which combines the desirable features of both rejection and MCMC sampling and thus removes the need to make a choice between the two. We call this algorithm the *Reject-Regenerate* sampler.

The Reject-Regenerate sampler has the following desirable features. At a given step t, using an exponentially tilted proposal density, the Reject-Regenerate sampler simulates a random variable $\boldsymbol{X}_t$. Then, with a certain probability the variable $\boldsymbol{X}_t$ is flagged as belonging to either one of these three states:

1. an draw within an Markov chain which initiates the next regenerative cycle;
2. an independent and exact/perfect draw from the target;
3. a regular draw within an MCMC run (which is neither exact, nor regenerative).

As a result of these features, the *Reject-Regenerate* sampler makes is unnecessary for the user to choose between rejection and MCMC sampling. If the rejection sampling is efficient, then most of the draws in the sequence $\{\boldsymbol{X}_t\}$ will be independent and exact draws from the target. However, if rejection sampling is not viable, then the sequence $\{\boldsymbol{X}_t\}$ will be interpreted as the output of an MCMC with the possibility of identifying regeneration cycles. In this way, the simulation effort in rejection sampling is recycled for MCMC sampling. In summary, we will describe a sampling scheme in which we identify the regeneration times of an independence sampler, and whenever regeneration occurs, it has a certain probability of achieving an independent exact draw from the target density. The regeneration is achieved using the classical method of "splitting" of the Markov transition kernel into mixture components, which was first proposed by Nummelin [18].

7.1 Nummelin Splitting of Transition Kernel

Suppose that the target pdf is $f(\boldsymbol{x}) = \frac{p(\boldsymbol{x})}{\ell}$ where $\ell = \int p(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$ is the normalizing constant and we have $p(\boldsymbol{x})$ available analytically. Our proposal pdf is $g(\boldsymbol{x})$ that satisfies $w(\boldsymbol{x}) := \frac{p(\boldsymbol{x})}{g(\boldsymbol{x})\exp(\psi^*)} \leq 1$, where $\psi^* = \max_{\boldsymbol{x}} \psi(\boldsymbol{x}) = \max_{\boldsymbol{x}} \ln \frac{p(\boldsymbol{x})}{g(\boldsymbol{x})}$. In the case where $g(\cdot;\boldsymbol{\mu})$ comes from a family of densities, indexed by some tilting parameter $\boldsymbol{\mu}$, we choose $\psi^* = \min_{\boldsymbol{\mu}} \max_{\boldsymbol{x}} \psi(\boldsymbol{x};\boldsymbol{\mu})$ and $g(\cdot;\boldsymbol{\mu}^*)$ is the corresponding optimal proposal.

Next denote $w_\rho(\boldsymbol{x}) := \min\{w(\boldsymbol{x})/\rho, 1\}$ for some $\rho \in (0,1]$. Now, suppose that we wish to simulate $\boldsymbol{X} \sim g$, conditional on $U \leq w_\rho(\boldsymbol{X})$. The probability of this happening is $c_\rho = \mathbb{E} w_\rho(\boldsymbol{X})$. The well-known rejection sampling corresponds to $\rho = 1$, giving acceptance probability $c_1 = \ell/\exp(\psi^*)$.

Recall that the probability transition kernel of an independence sampler with proposal g is

$$\kappa(\mathrm{d}\boldsymbol{y} \mid \boldsymbol{x}) = \alpha(\boldsymbol{y} \mid \boldsymbol{x}) g(\boldsymbol{y}) \mathrm{d}\boldsymbol{y} + (1 - \alpha^*(\boldsymbol{x}))\delta_{\boldsymbol{x}}(\mathrm{d}\boldsymbol{y})$$

where $\alpha(\boldsymbol{y} \mid \boldsymbol{x}) = 1 \vee \frac{w(\boldsymbol{y})}{w(\boldsymbol{x})}$, $\alpha^*(\boldsymbol{x}) = \int \alpha(\boldsymbol{u} \mid \boldsymbol{x}) g(\boldsymbol{u}) \, \mathrm{d}\boldsymbol{u}$. Given the current state of the Markov chain $\boldsymbol{x}$, the conventional implementation of the independence sampler draws $\boldsymbol{Y}' \sim g$, $U \sim \mathsf{Unif}(0, 1)$ and if $U < \frac{w(\boldsymbol{y}')}{w(\boldsymbol{x})}$, the next state of the chain $\boldsymbol{Y}$ is assigned $\boldsymbol{Y} \leftarrow \boldsymbol{Y}'$, otherwise $\boldsymbol{Y} \leftarrow \boldsymbol{x}$. This can be seen as drawing from the following transition kernel

$$\kappa(\mathrm{d}\boldsymbol{y}, \mathrm{d}\boldsymbol{y}', u \mid \boldsymbol{x}) = g(\boldsymbol{y}') \mathbb{1}_{u < \frac{w(\boldsymbol{y}')}{w(\boldsymbol{x})}} \delta_{\boldsymbol{y}'}(\mathrm{d}\boldsymbol{y}) \, \mathrm{d}\boldsymbol{y}' + g(\boldsymbol{y}') \mathbb{1}_{u > \frac{w(\boldsymbol{y}')}{w(\boldsymbol{x})}} \delta_{\boldsymbol{x}}(\mathrm{d}\boldsymbol{y}) \, \mathrm{d}\boldsymbol{y}'$$

which has the desired marginal $\kappa(\mathrm{d}\boldsymbol{y} \mid \boldsymbol{x})$ and marginals $\kappa(\mathrm{d}\boldsymbol{y}' \mid \boldsymbol{x}) = g(\mathrm{d}\boldsymbol{y}')$ and $\kappa(u \mid \boldsymbol{x}, \boldsymbol{y}') = 1, \ u \in (0, 1)$, so that

$$\kappa(\mathrm{d}\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{y}', u) = \mathbb{1}_{u < \frac{w(\boldsymbol{y}')}{w(\boldsymbol{x})}} \delta_{\boldsymbol{y}'}(\mathrm{d}\boldsymbol{y}) + \mathbb{1}_{u > \frac{w(\boldsymbol{y}')}{w(\boldsymbol{x})}} \delta_{\boldsymbol{x}}(\mathrm{d}\boldsymbol{y}).$$

Next, define $g_\rho(\boldsymbol{y}) := \frac{g(\boldsymbol{y}) w_\rho(\boldsymbol{y})}{c_\rho}$ and note that we have

$$1 \vee \frac{w(\boldsymbol{y})}{w(\boldsymbol{x})} \geq (1 \vee w(\boldsymbol{y})/\rho) \times (1 \vee \rho/w(\boldsymbol{x})) \geq (1 \vee w(\boldsymbol{y})/\rho) \times \rho,$$

and $\alpha^*(\boldsymbol{x}) \geq (1 \vee \rho/w(\boldsymbol{x})) \times c_\rho =: s_\rho(\boldsymbol{x})$. It follows that we can decompose $\kappa(\mathrm{d}\boldsymbol{y} \mid \boldsymbol{x})$ as a three-component mixture:

$$\begin{aligned} \kappa(\mathrm{d}\boldsymbol{y} \mid \boldsymbol{x}) = s_\rho(\boldsymbol{x}) g_\rho(\boldsymbol{y}) \, \mathrm{d}\boldsymbol{y} + (\alpha^*(\boldsymbol{x}) - s_\rho(\boldsymbol{x})) \frac{g(\boldsymbol{y})(1 \vee \frac{w(\boldsymbol{y})}{w(\boldsymbol{x})}) - g_\rho(\boldsymbol{y}) s_\rho(\boldsymbol{x})}{\alpha^*(\boldsymbol{x}) - s_\rho(\boldsymbol{x})} \mathrm{d}\boldsymbol{y} \\ + \big(1 - \alpha^*(\boldsymbol{x})\big) \delta_{\boldsymbol{x}}(\mathrm{d}\boldsymbol{y}). \end{aligned}$$

Regeneration happens whenever we simulate from the first component g_ρ (the idea being due to Nummelin [18]), again in practice this is done retrospectively. Formally, let us define

$$r(\boldsymbol{y} \mid \boldsymbol{x}) = \frac{(1 \vee w(\boldsymbol{y})/\rho) \times (1 \vee \rho/w(\boldsymbol{x}))}{1 \vee \frac{w(\boldsymbol{y})}{w(\boldsymbol{x})}} \leq 1. \tag{8}$$

Given previous state $\boldsymbol{x}$ and current state $\boldsymbol{Y}$ of the Markov chain, where $\boldsymbol{Y} \neq \boldsymbol{X}$ (that is, a transition has happened), one simulates another independent $V \sim \mathsf{Unif}(0, 1)$ and decides that $\boldsymbol{Y}$ initiates a new regenerative cycle if $V < r(\boldsymbol{Y} \mid \boldsymbol{X})$. In other words, one retrospectively identifies $\boldsymbol{Y}$ as a draw from g_ρ if $V < r(\boldsymbol{Y} \mid \boldsymbol{X})$. This is equivalent to sampling from the kernel:

$$\kappa(\mathrm{d}\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{y}', u, v) = \delta_{\boldsymbol{y}'}(\mathrm{d}\boldsymbol{y}) \mathbb{1}_{u < \frac{w(\boldsymbol{y}')}{w(\boldsymbol{x})}} \left[\mathbb{1}_{v < r(\boldsymbol{y}' \mid \boldsymbol{x})} + \mathbb{1}_{v > r(\boldsymbol{y}' \mid \boldsymbol{x})} \right] + \mathbb{1}_{u > \frac{w(\boldsymbol{y}')}{w(\boldsymbol{x})}} \delta_{\boldsymbol{x}}(\mathrm{d}\boldsymbol{y}).$$

Finally, to get the exact sampling as a subset of regeneration, define

$$e(\boldsymbol{y}) = \frac{w(\boldsymbol{y})}{1 \vee (w(\boldsymbol{y})/\rho)} \leq 1, \tag{9}$$

so that $c_\rho \geq c_1$. Then,

$$g_\rho(\boldsymbol{y}) = \frac{g(\boldsymbol{y})(1 \vee w(\boldsymbol{y})/\rho)}{c_\rho} = \frac{c_1}{c_\rho}\frac{g(\boldsymbol{y})w(\boldsymbol{y})}{c_1} + \left(1 - \frac{c_1}{c_\rho}\right)\frac{g(\boldsymbol{y})(1 \vee w(\boldsymbol{y})/\rho) - g(\boldsymbol{y})w(\boldsymbol{y})}{c_\rho - c_1}.$$

Notice that drawing from first component of this mixture $g(\boldsymbol{y})w(\boldsymbol{y}) \propto \pi(\boldsymbol{y})$ gives an exact draw from π. Simulation from this mixture is accomplished by sampling from the joint: $g_\rho(\boldsymbol{y}, v') = g_\rho(\boldsymbol{y})\mathbb{1}_{\{v' < e(\boldsymbol{y})\}} + g_\rho(\boldsymbol{y})\mathbb{1}_{\{v' > e(\boldsymbol{y})\}}$. In other words, simulate $\boldsymbol{Y} \sim g_\rho(\boldsymbol{y})$ and $V' \sim \mathsf{Unif}(0, 1)$ and then evaluate $\mathbb{1}_{\{V' < e(\boldsymbol{Y})\}}$ (to check if we sampled from the first component of this mixture).

Putting these observations together, we describe an algorithm where we simulate $\boldsymbol{Y} \sim g(\boldsymbol{y})$, independently $V, V', U \sim_{iid} \mathsf{Unif}(0, 1)$, and sample from $\kappa(\mathrm{d}\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{y}', u, v, v') =:$

$$\delta_{\boldsymbol{y}'}(\mathrm{d}\boldsymbol{y})\mathbb{1}_{u < \frac{w(\boldsymbol{y}')}{w(\boldsymbol{x})}}\left[\mathbb{1}_{v < r(\boldsymbol{y}' \mid \boldsymbol{x}), v' < e(\boldsymbol{y}')} + \mathbb{1}_{v < r(\boldsymbol{y}' \mid \boldsymbol{x}), v' > e(\boldsymbol{y}')} + \mathbb{1}_{v > r(\boldsymbol{y}' \mid \boldsymbol{x})}\right] + \mathbb{1}_{u > \frac{w(\boldsymbol{y}')}{w(\boldsymbol{x})}}\delta_{\boldsymbol{x}}(\mathrm{d}\boldsymbol{y}).$$

The probability of exact sampling, conditional on $\boldsymbol{x}$ is: $s_\rho(\boldsymbol{x}) \times \frac{c_1}{c_\rho} = c_1 \times (1 \vee \rho/w(\boldsymbol{x}))$. The final algorithm is thus as follows (here $B = 1$ means regenerative draw and $B = 2$ means exact sampling draw).

Algorithm 2: MCMC with regeneration and exact sampling

Input: Current state of chain $(\boldsymbol{X}_n, B_n)$ and constant ρ.

$B_{n+1} \leftarrow 0$, simulate $\boldsymbol{Y} \sim g(\boldsymbol{y})$ and $U, V, V' \sim_{\text{iid}} \mathsf{Unif}(0, 1)$, independently;
if $U \leq w(\boldsymbol{Y})/w(\boldsymbol{X}_n)$ **then**
  $\boldsymbol{X}_{n+1} \leftarrow \boldsymbol{Y}$;
  if $V \leq r(\boldsymbol{Y} \mid \boldsymbol{X}_n)$ *as in* (8) **then**
    $B_{n+1} \leftarrow 1$;
    if $V' \leq e(\boldsymbol{Y})$ *as in* (9) **then**
      $B_{n+1} \leftarrow 2$;
else
  $\boldsymbol{X}_{n+1} \leftarrow \boldsymbol{X}_n$;
return $(\boldsymbol{X}_{n+1}, B_{n+1})$ *as the next state of the chain*

The special case of $\rho = 1$ corresponds to an independence sampler with regeneration, but with each regeneration corresponding to an exact draw from π. In other words, every regenerative cycle is initialized by an exact draw from the target.

Applying Algorithm 2 on the Women's Wages dataset from Sect. 5, we find that when $\rho = 0.5c = 0.5\psi(\boldsymbol{z}^*, r^*, \boldsymbol{\mu}^*, \eta^*)$, the frequency of regenerative outcomes is observed to be 0.45, while the frequency of exact draws is 0.29. Here we do not address the interesting question of choosing the constant ρ optimally.

7.2 Rare-Event Robustness

Of interest is the robustness of Algorithm 2 in the case when the target is the pdf of $\boldsymbol{Y} \sim \mathsf{t}_\nu(\mathbf{0}, \Sigma)$, conditional on $\{\boldsymbol{Y} \geq \boldsymbol{l}(\gamma)\}$, as the rarity parameter γ diverges to infinity.

The concept of rare-event efficiency for MCMC sampling was introduced in [4]. Briefly, if $\kappa_t(A|\boldsymbol{x})$ is the t-step transition kernel of a Markov chain with limiting and stationary density f, then the total variation distance is $D_t(\boldsymbol{x}) = \sup_A |\kappa_t(A|\boldsymbol{x}) - f(A)|$, where $f(A) := \int_A f(\boldsymbol{x})\mathrm{d}\boldsymbol{x}$ is the measure of a Borel set A on $\mathbb{R}^d$. Taking at least $T = \min\{t : D_t(\boldsymbol{x}) \leq \epsilon\}$ MCMC number of steps will keep the total variation distance below ϵ. A Markov chain is *strongly efficient* if $\limsup_\gamma T(\gamma) < \infty$ and *logarithmically efficient* if $\limsup_\gamma \frac{\ln T(\gamma)}{\ln \gamma} < \infty$. Note that the length of a strongly efficient chain does not have to be increased as the rarity parameter γ gets larger and larger. But the length of a logarithmically efficient chain must grow at a polynomial rate to ensure that the total variation distance stays below ϵ.

Just like the exponentially tilted estimator $\hat{\ell}$ is strongly efficient as per Theorem 3, the independence sampler (and hence the Reject-Regenerate Algorithm 2) is also strongly efficient for sampling $\boldsymbol{Y} \sim \mathsf{t}_\nu(\mathbf{0}, \Sigma)$, conditional on $\{\boldsymbol{Y} \geq \boldsymbol{l}(\gamma)\}$, when we apply the exponentially tilted sequential proposal density (2). In other words, we have the following theorem whose proof is given in the Appendix.

Theorem 4 (Strongly Efficient Reject-Regenerate Sampling) *As $\gamma \uparrow \infty$, the Reject-Regenerate Algorithm 2 using the optimal exponentially tilted sequential proposal density* (2) *simulates a strongly efficient Markov chain with target pdf* (1).

8 Concluding Remarks

In this paper we establish the bounded relative error of the IS estimator $\hat{\ell} = \exp(\psi(\boldsymbol{Z}, R; \boldsymbol{\mu}^*, \eta^*))$ in a rare-event regime. A byproduct of this proof is a multivariate extension of the Mill's ratio, currently known only for univariate student densities.

We describe novel applications of rejection-sampling Algorithm 1 on the Bayesian inference for: (a) the constrained linear regression model; (b) the Tobit model; (c) the non-negative smoothing spline model.

We have also tested these rejection samplers on real and synthetic datasets. Our simulation experience reveals that these samplers achieve valid posterior inferences and the probabilities of retaining samples are reasonably high.

We have also proposed a new sampler, which we call the Reject-Regenerate sampler. The proposed algorithm identifies regeneration times within the Markov chain, and in the event of a regeneration, with some probability, the Markov chain achieves an exact draw from the target. The validity of this sampler is established by rewriting its transition kernel as a nested mixture with a regenerative and non-regenerative

components. Whenever a draw is made from the regenerative component, it initializes a new regenerative cycle. We further decompose the regenerative component into a mixture that includes the target density. In this manner we have an independence sampler whose regeneration times can be identified, and whenever a new regenerative cycle is initialized, there is chance that the cycle starts with an exact draw from the target density.

Finally, we establish that the version of the Reject-Regenerate sampler using an exponentially tilted proposal density is asymptotically strongly efficient in a rare-event setting.

Appendix

Proof of Theorem 2

Proof First, we use the normal scale-mixture representation of $\boldsymbol{Y} \sim \mathsf{t}_\nu(\mathbf{0}, \Sigma)$ as $\boldsymbol{Y} = \sqrt{\nu}\boldsymbol{Z}/R$, where $\boldsymbol{Z} \sim \mathcal{N}(\mathbf{0}, \Sigma)$ is independent of $R \sim c_\nu(r) = \frac{\exp\left(-\frac{r^2}{2}+(\nu-1)\ln r\right)}{2^{\nu/2-1}\Gamma(\nu/2)}$, $r > 0$. We can thus write ℓ as a conditional expectation: $\ell(\gamma) = \mathbb{P}\left[\frac{\sqrt{\nu}\boldsymbol{Z}}{R} \geq \boldsymbol{l}(\gamma)\right] = \mathbb{E}\left[\mathbb{P}\left[\frac{\sqrt{\nu}\boldsymbol{Z}}{R} \geq \boldsymbol{l}(\gamma) \,\middle|\, R\right]\right]$. Next, condition on $R = r$, and let $\boldsymbol{\mu} = r\boldsymbol{x}^*/\sqrt{\nu}$, where $\boldsymbol{x}^*$ is the solution of the QPP. Denoting $\boldsymbol{t} = [\boldsymbol{t}_1^\top, \boldsymbol{t}_2^\top]^\top =: r\boldsymbol{l}/\sqrt{\nu}$, and making a change of variable $\boldsymbol{z} \leftarrow \boldsymbol{z} - \boldsymbol{\mu}$, we obtain $\mathbb{P}\left[\frac{\sqrt{\nu}\boldsymbol{Z}}{R} \geq \boldsymbol{l}(\gamma) \,\middle|\, R = r\right] = \mathbb{P}[\boldsymbol{Z} \geq \boldsymbol{t}] =$

$$
\begin{aligned}
&= \mathbb{E}\exp(-\frac{\boldsymbol{\mu}^\top\Sigma^{-1}\boldsymbol{\mu}}{2} - \boldsymbol{Z}^\top\Sigma^{-1}\boldsymbol{\mu})\mathbb{1}\{\boldsymbol{Z} \geq \boldsymbol{t} - \boldsymbol{\mu}\} \\
&= \exp(-\frac{\boldsymbol{\mu}^\top\Sigma^{-1}\boldsymbol{\mu}}{2})\mathbb{E}\exp(-\boldsymbol{Z}_1^\top\Sigma_{11}^{-1}\boldsymbol{t}_1)\mathbb{1}\{\boldsymbol{Z}_1 \geq \boldsymbol{t}_1 - \boldsymbol{\mu}_1, \boldsymbol{Z}_2 \geq \boldsymbol{t}_2 - \boldsymbol{\mu}_2\} \\
&= \exp(-\boldsymbol{t}_1^\top\Sigma_{11}^{-1}\boldsymbol{t}_1/2)\mathbb{E}\exp(-\boldsymbol{Z}_1^\top\Sigma_{11}^{-1}\boldsymbol{t}_1)\mathbb{1}\{\boldsymbol{Z}_1 \geq \mathbf{0}, \boldsymbol{Z}_2 \geq \boldsymbol{t}_2 - \boldsymbol{\mu}_2\}.
\end{aligned}
$$

In other words, we have:

$$
\mathbb{P}[\boldsymbol{Z} \geq \boldsymbol{t}] = \exp(-\frac{r^2\boldsymbol{l}_1^\top\Sigma_{11}^{-1}\boldsymbol{l}_1}{2\nu})\mathbb{E}\exp(-\frac{r\boldsymbol{Z}_1^\top\Sigma_{11}^{-1}\boldsymbol{l}_1}{\sqrt{\nu}})\mathbb{1}\{\boldsymbol{Z}_1 \geq \mathbf{0}, \boldsymbol{Z}_2 \geq \frac{r(\boldsymbol{l}_2 - \Sigma_{21}\Sigma_{11}^{-1}\boldsymbol{l}_1)}{\sqrt{\nu}}\} \tag{10}
$$

Let $\mathfrak{D} \equiv \{\boldsymbol{z} : \boldsymbol{z}_1 \geq \mathbf{0}, \boldsymbol{z}_2 \geq \frac{r(\boldsymbol{l}_2-\Sigma_{21}\Sigma_{11}^{-1}\boldsymbol{l}_1)}{\sqrt{\nu}}\}$. We can now rewrite (10) as an integral and integrate over r. This gives $\ell(\gamma) =:$

$$
\begin{aligned}
&= \int_0^\infty \int_{\mathfrak{D}} c_\nu(r)\phi_\Sigma(\boldsymbol{z}) \exp\left(-r^2 \boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1/(2\nu) - r\boldsymbol{z}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1/\sqrt{\nu}\right) \mathrm{d}\boldsymbol{z}\mathrm{d}r \\
&= \frac{2^{1-(\nu+d)/2}\pi^{-d/2}}{\Gamma(\frac{\nu}{2})|\Sigma|^{1/2}} \int_0^\infty \int_{\mathfrak{D}} \exp\left(-\frac{r^2}{2}\left(1+\frac{\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{2\nu}\right) - \frac{\boldsymbol{z}^\top \Sigma^{-1}\boldsymbol{z}}{2} - \frac{r\boldsymbol{z}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{\sqrt{\nu}} + (\nu-1)\ln r\right) \mathrm{d}\boldsymbol{z}\mathrm{d}r \\
&= \frac{2^{1-(\nu+d)/2}\pi^{-d/2}}{\Gamma(\frac{\nu}{2})|\Sigma|^{1/2}\left(1+\frac{\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{\nu}\right)^{\nu/2}} \int_0^\infty \int_{\mathfrak{D}} \exp\left(-\frac{u^2}{2} - \frac{\boldsymbol{z}^\top \Sigma^{-1}\boldsymbol{z}}{2} - \frac{u\,\boldsymbol{z}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{\sqrt{\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}} + (\nu-1)\ln u\right) \mathrm{d}\boldsymbol{z}\mathrm{d}u \\
&= \frac{1}{\left(1+\frac{\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{\nu}\right)^{\nu/2}} \int_0^\infty \int_{\mathbb{R}^d} c_\nu(u)\phi_\Sigma(\boldsymbol{z}) \exp\left(-\frac{u\,\boldsymbol{z}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{\sqrt{\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}}\right) \mathbb{1}\left\{\boldsymbol{z}_1 \geq \boldsymbol{0}, \boldsymbol{z}_2 \geq \frac{u(\boldsymbol{l}_2-\Sigma_{21}\Sigma_{11}^{-1}\boldsymbol{l}_1)}{\sqrt{\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}}\right\} \mathrm{d}\boldsymbol{z}\mathrm{d}u \\
&= \left(1+\frac{\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{\nu}\right)^{-\nu/2} \mathbb{E}\exp\left(-\frac{R\,\boldsymbol{Z}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{\sqrt{\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}}\right) \mathbb{1}\left\{\boldsymbol{Z}_1 \geq \boldsymbol{0}, \boldsymbol{Z}_2 \geq \frac{R(\boldsymbol{l}_2-\Sigma_{21}\Sigma_{11}^{-1}\boldsymbol{l}_1)}{\sqrt{\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}}\right\},
\end{aligned}
$$

where the third line follows from the change of variable $u = r\sqrt{1+\frac{\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{\nu}}$. Next, using formula (10) we rewrite the last expression as:

$$
\left(1+\frac{\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{\nu}\right)^{-\nu/2} \mathbb{E}\exp\left(\frac{R^2\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{2(\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1)}\right) \mathbb{P}\left[\boldsymbol{Z} \geq \frac{R\boldsymbol{l}}{\sqrt{\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}} \,\middle|\, R\right].
$$

We now seek to apply the dominated convergence theorem to the expectation in the last displayed equation. For this we need the upper bound (recall that $\Sigma_{11}^{-1}\boldsymbol{l}_1 \geq \boldsymbol{0}$)

$$
\begin{aligned}
\exp\left(\frac{r^2\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{2(\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1)}\right) \mathbb{P}\left[\boldsymbol{Z} \geq \frac{r\boldsymbol{l}}{\sqrt{\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}}\right] &\leq \exp(r^2/2)\mathbb{P}\left[\boldsymbol{Z}_1 \geq \frac{r\boldsymbol{l}_1}{\sqrt{\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}}\right] \\
&\leq \exp(r^2/2)\mathbb{P}\left[\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{Z}_1 \geq \frac{r\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{\sqrt{\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}}\right] \\
&= \exp(r^2/2)\overline{\Phi}\left[r\sqrt{\frac{\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}}\right] \leq \exp(r^2/2)\overline{\Phi}(r).
\end{aligned}
$$

The last expression is integrable in the sense that $\int_0^\infty c_\nu(r)\exp(r^2/2)\overline{\Phi}(r)\,\mathrm{d}r =$

$$
\frac{2^{1-\nu/2}}{\Gamma(\nu/2)}\int_0^\infty r^{\nu-1}\overline{\Phi}(r)\,\mathrm{d}r = \frac{2^{1-\nu/2}}{\Gamma(\nu/2)2\nu}\int_{-\infty}^\infty |u|^\nu \phi(u)\mathrm{d}u = \frac{2^{1-\nu/2}\Gamma((\nu+1)/2)2^{\nu/2}}{\sqrt{\pi}\Gamma(\nu/2)2\nu} = \frac{\Gamma((\nu+1)/2)}{\sqrt{\pi}\Gamma(\nu/2)\nu} < \infty.
$$

In addition, as $\gamma \uparrow \infty$, by Lemma 1 we have the pointwise limits:

$$
\exp\left[\frac{r^2\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{2(\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1)}\right] \mathbb{P}\left(\boldsymbol{Z} \geq \frac{r\boldsymbol{l}}{\sqrt{\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}}\right) \to \exp(r^2/2)\mathbb{P}[\boldsymbol{Z} \geq r\boldsymbol{l}_\infty].
$$

Therefore, by the dominated convergence theorem

$$\lim_{\gamma\uparrow\infty} \mathbb{E}\exp\left(\frac{R^2 \boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}{2(\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1)}\right)\mathbb{P}\left[\boldsymbol{Z}\geq \frac{R\boldsymbol{l}}{\sqrt{\nu+\boldsymbol{l}_1^\top \Sigma_{11}^{-1}\boldsymbol{l}_1}}\,\middle|\, R\right] = \frac{2^{1-\nu/2}}{\Gamma(\nu/2)}\int_0^\infty r^{\nu-1}\mathbb{P}[\boldsymbol{Z}\geq r\boldsymbol{l}_\infty]\mathrm{d}r.$$

This concludes the proof. □

Lemma 1 (Continuity of Gaussian tail) *Suppose that* $\boldsymbol{Z}\sim\mathcal{N}(\boldsymbol{0},\,\Sigma)$ *for some positive definite matrix* Σ*, and* $\boldsymbol{a}_n\to\boldsymbol{a}$ *as* $n\uparrow\infty$*. Then, the tail of the multivariate Gaussian is continuous:* $\lim_{n\uparrow\infty}\mathbb{P}[\boldsymbol{Z}\geq\boldsymbol{a}_n]=\mathbb{P}[\boldsymbol{Z}\geq\boldsymbol{a}]$.

Proof The proof is yet another application of the dominated convergence theorem to show that: $\int_{[\boldsymbol{0},\infty)}\phi_\Sigma(z+\boldsymbol{a}_n)\mathrm{d}z\to\int_{[\boldsymbol{0},\infty)}\phi_\Sigma(z+\boldsymbol{a})\mathrm{d}z=\mathbb{P}[\boldsymbol{Z}\geq\boldsymbol{a}]$. Since Σ is a positive definite matrix, the $\|\boldsymbol{x}\|_\Sigma^2:=\boldsymbol{x}^\top\Sigma^{-1}\boldsymbol{x}$ is a norm satisfying $\|z+\boldsymbol{a}_n\|_\Sigma^2\leq 2(\|z\|_\Sigma^2+\|\boldsymbol{a}_n\|_\Sigma^2)$. Therefore, $\int_{[\boldsymbol{0},\infty)}\phi_\Sigma(z+\boldsymbol{a}_n)\mathrm{d}z\leq\frac{\exp(-\|\boldsymbol{a}_n\|_\Sigma^2)}{2^{n/2}}\int_{[\boldsymbol{0},\infty)}\phi_{\Sigma/2}(z)\mathrm{d}z<\infty$, and the conditions for the dominated convergence theorem are met. □

Proof of Theorem 3

Proof First, note that the second moment is $\int g(z,r;\boldsymbol{\mu}^*,\eta^*)\exp(2\psi(z,r;\boldsymbol{\mu}^*,\eta^*))\,\mathrm{d}z\mathrm{d}r=$

$$=\int_{\mathfrak{R}} c_\nu(r)\phi_\Sigma(z)\exp(\psi(z,r;\boldsymbol{\mu}^*,\eta^*))\mathrm{d}z\mathrm{d}r\leq\ell(\gamma)\exp(\psi(z^*,r^*;\boldsymbol{\mu}^*,\eta^*)).$$

Since the properties of ψ imply that

$$\psi(z^*,r^*;\boldsymbol{\mu}^*,\eta^*)\leq\psi(z^*,r^*;\boldsymbol{0},\eta^*)\leq\frac{(\eta^*)^2}{2}-r^*\eta^*+(\nu-1)\ln r^*+\ln\Phi(\eta^*),$$

bounded relative error will follow if we can show that $\frac{(r^*)^{\nu-1}\Phi(\eta^*)\exp(\frac{(\eta^*)^2}{2}-r^*\eta^*)}{\ell(\gamma)}$ remains bounded in γ. The pair (r^*,η^*) is determined from the solution to (3), namely from finding the saddle-point solution of: $\max_{r,z}\min_{\eta,\boldsymbol{\mu}}\psi(z,r;\boldsymbol{\mu},\eta)$. This can be obtained by setting the gradient of ψ with respect to the vector $(z,r,\boldsymbol{\mu},\eta)$ to zero: $\nabla\psi=\boldsymbol{0}$. We now introduce the following notation that will allow us to express $\nabla\psi=\boldsymbol{0}$ explicitly. Let L be the lower triangular Cholesky factor of $\Sigma=LL^\top$. Define $D=\mathrm{diag}(L)$, $\breve{L}=D^{-1}L$, $\tilde{\boldsymbol{l}}=\frac{r}{\sqrt{\nu}}D^{-1}\boldsymbol{l}(\gamma)-(\breve{L}-I)z$, and vector $\boldsymbol{\Psi}$ with elements $\Psi_k=\phi(\tilde{l}_k-\mu_k)/\overline{\Phi}(\tilde{l}_k-\mu_k)$. Then, $\nabla\psi=\boldsymbol{0}$ can be written as

$$\begin{aligned}(\check{L}^\top - I)\boldsymbol{\Psi} - \boldsymbol{\mu} &= \mathbf{0}\\ \frac{\nu-1}{r} - \eta - \frac{1}{\sqrt{\nu}}\boldsymbol{\Psi}^\top D^{-1}\boldsymbol{l}(\gamma) &= 0\\ \boldsymbol{\mu} + \boldsymbol{\Psi} - \boldsymbol{z} &= \mathbf{0}\\ \eta + \frac{\phi(\eta)}{\Phi(\eta)} - r &= 0.\end{aligned} \tag{11}$$

Next, we verify via substitution that the solution of (11) as $\gamma \uparrow \infty$ satisfies $r^* = \mathcal{O}(\gamma^{-1})$, $\boldsymbol{z}^* = \mathcal{O}(\mathbf{1})$, $\eta^* = \mathcal{O}(-\gamma)$, $\boldsymbol{\mu}^* = \mathcal{O}(\mathbf{1})$. First, equations one and three in (11) are trivially satisfied and we can deduce that $\boldsymbol{\Psi} = \mathcal{O}(\mathbf{1})$. Second, since $\tilde{\boldsymbol{l}} = \mathcal{O}(r\boldsymbol{l}(\gamma)) = \mathcal{O}(\mathbf{1})$, it follows that equation two in (11) is equivalent to

$$r^*\eta^* = \nu - 1 - \frac{r^*}{\sqrt{\nu}}\boldsymbol{\Psi}^\top D^{-1}\boldsymbol{l}(\gamma) = \mathcal{O}(1).$$

Finally, note that Mill's ratio $\frac{\Phi(\eta)}{\phi(\eta)} \simeq -\frac{1}{\eta} + \frac{1}{\eta^3}$, $\eta \downarrow -\infty$, implies that equation four is asymptotically equivalent to $r\eta^2 + \eta - r \simeq 0$. The solution of this quadratic equation in turn implies that $\eta \simeq (-1 - \sqrt{1+4r^2})/(2r) \simeq -1/r$. In other words, $\eta^* r^* = \mathcal{O}(1)$, as desired. Therefore, if $\tilde{\psi}$ denotes the value of ψ at the solution (11), we have

$$\begin{aligned}\tilde{\psi} &= \frac{\|\boldsymbol{\mu}^*\|^2}{2} - (\boldsymbol{z}^*)^\top\boldsymbol{\mu}^* + \frac{(\eta^*)^2}{2} - r^*\eta^* + (\nu-1)\ln r^* + \ln\Phi(\eta^*) + \sum_{k=1}^{d}\ln\overline{\Phi}(\tilde{l}_k - \mu_k^*)\\ &= \mathcal{O}(1) + \frac{(\eta^*)^2}{2} + (\nu-1)\ln r^* + \ln\overline{\Phi}(-\eta^*).\end{aligned}$$

By Mill's ratio inequality: $\ln\overline{\Phi}(-\eta) \le -\eta^2/2 - \frac{1}{2}\ln(2\pi) - \ln(-\eta)$, we obtain: $\tilde{\psi} \lesssim \mathcal{O}(1) - \ln(-\eta^*) - \frac{1}{2}\ln(2\pi) + (\nu-1)\ln r^* = -\nu\log(\gamma) + \mathcal{O}(1)$. In other words, there exist constants $c_1, c_2 > 0$ such that $\exp(\tilde{\psi}) \le c_1\gamma^{-\nu}$ for every $\gamma > c_2$. Therefore,

$$\mathrm{Var}(\hat{\ell}) = \mathbb{E}\exp(\psi(\boldsymbol{Z}, R; \boldsymbol{\mu}^*, \eta^*)) - \ell^2 \lesssim \ell(\gamma)\exp(\tilde{\psi}) - \ell^2 \le c_1\gamma^{-2\nu} - \ell^2(\gamma)$$

and since by Theorem 2

$$\ell(\gamma) \simeq c \times \left(1 + \gamma^2 \times \underbrace{\frac{\boldsymbol{l}_1^\top\Sigma_{11}^{-1}\boldsymbol{l}_1}{\nu\times\gamma^2}}_{\Theta(1)}\right)^{-\nu/2} = \Theta(\gamma^{-\nu}), \quad \gamma \uparrow \infty,$$

we have $\limsup_{\gamma\uparrow\infty}\mathrm{Var}(\hat{\ell})/\ell^2 < \infty$. □

Proof of Theorem 4

Proof Ignoring the B_i variable in Algorithm 2 gives a state $\boldsymbol{X}_n$ with marginal distribution that follows an independence Metropolis Hastings sampler. From [15, Theorem 2.1] we know that for an independence Metropolis sampler with proposal $g(\boldsymbol{x})$ and target $f(\boldsymbol{x})$ such that $\sup_{\boldsymbol{x}} f(\boldsymbol{x})/g(\boldsymbol{x}) < c$ for some constant $c > 0$, the Markov chain is uniformly ergodic with convergence rate

$$\sup_A |\kappa_t(A|\boldsymbol{x}) - f(A)| \leq (1 - c^{-1})^t.$$

Thus, to ensure the total variation bound remains below ϵ, we need to run the independence sampler for t^* steps such that

$$(1 - c^{-1})^{t^*} \leq \exp(-t^*/c) \leq \epsilon.$$

In other words, we have $t^* \geq \lceil -c \ln(\epsilon) \rceil$ and the length of the chain will remain bounded in the rarity parameter γ provided that $c(\gamma)$ remains bounded in γ. In Algorithm 2 we have

$$c(\gamma) = f(\boldsymbol{x})/g(\boldsymbol{x}) = \frac{p(\boldsymbol{x})}{g(\boldsymbol{x})\ell(\gamma)} \leq \frac{\exp(\psi^*)}{\ell(\gamma)} \leq \frac{(r^*)^{\nu-1}\Phi(\eta^*)\exp(\frac{(\eta^*)^2}{2} - r^*\eta^*)}{\ell(\gamma)},$$

where $\psi^* = \exp(\psi(\boldsymbol{z}^*, r^*; \boldsymbol{\mu}^*, \eta^*))$. However, from the proof of Theorem 3 we know that $\frac{\exp(\psi^*)}{\ell(\gamma)}$ remains bounded as $\gamma \uparrow \infty$. Hence, the Markov chain in Algorithm 2 is strongly efficient. □

References

1. Botev, Z.I.: The normal law under linear restrictions: simulation and estimation via minimax tilting. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **79**(1), 125–148 (2017)
2. Botev, Z.I., L'Ecuyer, P.: Efficient probability estimation and simulation of the truncated multivariate student-t distribution. In: 2015 Winter Simulation Conference (WSC), pp. 380–391. IEEE (2015)
3. Botev, Z., L'Ecuyer, P.: Simulation from the normal distribution truncated to an interval in the tail. In: proceedings of the 10th EAI International Conference on Performance Evaluation Methodologies and Tools on 10th EAI International Conference on Performance Evaluation Methodologies and Tools, pp. 23–29 (2017)
4. Botev, Z.I., Mackinlay, D., Chen, Y.L.: Logarithmically efficient estimation of the tail of the multivariate normal distribution. In: 2017 Winter Simulation Conference (WSC), pp. 1903–1913. IEEE (2017)
5. Botev, Z.I., Chen, Y.L., L'Ecuyer, P., MacNamara, S., Kroese, D.P.: Exact posterior simulation from the linear lasso regression. In: 2018 Winter Simulation Conference (WSC), pp. 1706–1717. IEEE (2018)
6. Chen, M.H., Deely, J.J.: Bayesian analysis for a constrained linear multiple regression problem for predicting the new crop of apples. J. Agric. Biol. Environ. Stat. **1**(4), 467–489 (1996)

7. Chen, M.H., Ibrahim, J.G., Shao, Q.M.: Monte Carlo Methods in Bayesian Computation. Springer (2000)
8. Chib, S.: Bayes inference in the Tobit censored regression model. J. Econom. **51**(1–2), 79–99 (1992)
9. Gelfand, A.E., Smith, A.F., Lee, T.M.: Bayesian analysis of constrained parameter and truncated data problems using Gibbs sampling. J. Am. Stat. Assoc. **87**(418), 523–532 (1992)
10. Genz, A., Bretz, F.: Numerical computation of multivariate t-probabilities with application to power calculation of multiple contrasts. J. Stat. Comput. Simul. **63**(4), 103–117 (1999)
11. Hashorva, E., Hüsler, J.: On multivariate Gaussian tails. Ann. Inst. Stat. Math. **55**(3), 507–522 (2003)
12. Kroese, D.P., Botev, Z.I., Taimre, T., Vaisman, R.: Data Science and Machine Learning: Mathematical and Statistical Methods. Chapman and Hall/CRC (2019)
13. Kroese, D.P., Taimre, T., Botev, Z.I.: Handbook of Monte Carlo Methods. Wiley (2011)
14. L'Ecuyer, P., Blanchet, J.H., Tuffin, B., Glynn, P.W.: Asymptotic robustness of estimators in rare-event simulation. ACM Trans. Model. Comput. Simul. (TOMACS) **20**(1), 1–41 (2010)
15. Mengersen, K.L., Tweedie, R.L.: Rates of convergence of the Hastings and Metropolis algorithms. Ann. Stat. **24**(1), 101–121 (1996)
16. Mills, J.P.: Table of the ratio: area to bounding ordinate, for any portion of normal curve. Biometrika, pp. 395–400 (1926)
17. Mroz, T.A.: The sensitivity of an empirical model of married women's hours of work to economic and statistical assumptions. Econom. J. Econom. Soc. **55**(4), 765–799 (1987)
18. Nummelin, E.: A splitting technique for Harris recurrent Markov chains. Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete **43**(4), 309–318 (1978)
19. Pakman, A., Paninski, L.: Exact Hamiltonian Monte Carlo for truncated multivariate Gaussians. J. Comput. Graph. Stat. **23**(2), 518–542 (2014)
20. Soms, A.P.: An asymptotic expansion for the tail area of the t-distribution. J. Am. Stat. Assoc. **71**(355), 728–730 (1976)
21. Soms, A.P.: Rational bounds for the t-tail area. J. Am. Stat. Assoc. **75**(370), 438–440 (1980)

Quasi-Monte Carlo Methods in Portfolio Selection with Many Constraints

Alexander Brunhuemer and Gerhard Larcher

Abstract We describe a concrete on-going industry project on advanced portfolio optimization based on machine-learning techniques, and we report on attempts and results of successful and advantageous application of QMC methods in this project. We are also carrying out an approach to determine a measure for dispersion in an opportunity set, which cannot trivially be found, because of the uncertainty of the shape of an opportunity set. Finally, we state some still open problems and questions in this context.

Keywords Portfolio-optimization · Quasi-Monte Carlo methods · Dispersion

1 Introduction

We all know that it is a strong concern and a great strength of Pierre L'Ecuyer to provide powerful and user-optimized algorithms and also software for the application of QMC methods to a broad audience of potential users, both in academics as well as in industry.

I remember very well: In 2018, when we applied for the continuation of a very large research project ("Special Research Area (SFB): Quasi-Monte Carlo Methods: Theory and Applications") of the Austrian Science Fund, that had started in 2014, Pierre was one of the reviewers, and he was a member of the jury at the corresponding scientific hearing in Vienna. In one of his statements at this hearing, Pierre proposed, that in our work in the SFB accompanying our research work we could intensify our efforts in the direction of QMC software development. This proposal by Pierre was one of the incentives for the creation of our working group LSQF (Linz School of Quantitative Finance, www.lsqf.org) in which we carry out MC- and QMC-based

A. Brunhuemer · G. Larcher (✉)
Institute for Financial Mathematics and Applied Number Theory, JKU Linz, Linz, Austria
e-mail: gerhard.larcher@jku.at

A. Brunhuemer
e-mail: alexander.brunhuemer@jku.at

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_5

industry projects in the field of quantitative finance and in which we have developed corresponding open source MC- and QMC-based quantitative finance software.

To be honest: Many real-world applications (especially in quantitative finance), which rely on simulation techniques, can be handled and managed successfully with the help of pure MC methods. If we compare the results of these MC simulations with the results we obtain when we use QMC techniques (i.e., roughly speaking, when we use carefully chosen extremely well distributed simulation samples instead of (pseudo-)random samples) then often these QMC approaches do not provide better, or only slightly better (at the cost of higher computation time) results than pure Monte Carlo. A survey of some such industry-projects carried out via LSQF can be found in Chap. 10 of the monograph [6].

Sometimes, however, we have to deal with challenges where QMC indeed can improve the performance of our approaches considerably. In this paper we will give such an example of a concrete industry project, where we strongly believe that we will be able to successfully apply QMC methods. This is still ongoing work, and we will state and explain here the problem and present first investigations and results in this direction. The basic topic of the project is advanced portfolio optimization with many constraints, based also on machine learning techniques.

Our investigations are obviously not the first ones where MC (or QMC) simulations are used in portfolio optimization or asset allocation (see e.g. [1–3, 11]). These papers follow a more general approach in terms of the considered utility function where we restrict ourselves to the classical Sharpe ratio. However, our approach could easily be extended to general utility functions. Additionally, we are not exclusively interested in finding the optimal portfolio, but are highly interested in the coverage of the opportunity set. This should provide insights into the quality of our approach when looking at portfolio optimization with many (and especially more complex) constraints.

Principal Remarks

We are very well aware of two facts: First, there are reservations against a use of classical portfolio optimization theory of Markowitz. However, we explicitly were instructed by our industry partner to implement an adapted version of this Markowitz portfolio selection system. Thus, for us it was not in question to discuss pros and cons of this theory and, therefore, this is also not a topic in this paper.

Secondly, as will be also addressed several times later in this paper, we know very well that there exist a number of other powerful techniques (apart from MC- and QMC-simulation) to solve the challenges of Markowitz portfolio selection very efficiently. For a survey see e.g. the paper [10]. However, since in our concrete industry project we had to deal with many additional constraints (see Sect. 3 for some details), and were required to deliver a solution with great flexibility (in terms of expandability as well as fast generation of multiple variants) while staying in a tight financial budget, we restricted our approach to MC- and QMC simulation. For this reason, we also restrict our investigations in this paper to MC- and QMC approaches.

2 Classical Portfolio Selection in a Nutshell

We start with giving a very short introduction to classical portfolio selection theory which was founded by Harry Markowitz in the 1950s [8]. The setting is the following:

- We should (now, i.e. at time 0) create a financial portfolio and we want to hold this portfolio until time T in the future.
- We have a certain amount of money in our domestic currency to invest. For simplicity we say the investment is 1 Euro.
- We have s assets $A_1, A_2, \ldots, A_s$ in which we are allowed to invest our money.
- For each of these A_i we assume that we have given an estimate μ_i for its expected per annum return (given in percent) in the time interval $[0, T]$.
- For each of these A_i we assume that we have given an estimate σ_i for its expected per annum volatility (given in percent), i.e., for the standard deviation of its returns in the time interval $[0, T]$.
- For each pair (i, j) with $1 \le i \ne j \le s$ we assume that we have given an estimate ρ_{ij} for the correlation between returns of asset A_i and asset A_j in the time interval $[0, T]$. We set $\rho_{ii} := 1$ and denote by C the correlation matrix $(\rho_{ij})_{i,j=1,\ldots,s}$.
- We build portfolios P which we denote by $x_1A_1 + x_2A_2 + \cdots + x_sA_s$ with $x_1 + x_2 + \cdots + x_s = 1$, which means: Invest x_i Euro of your money in asset A_i. Here we restrict to non-negative weights x_i (meaning, that no short selling of assets is allowed).

We obviously have for the expected return $\mu(P) = \mu(x_1, \ldots, x_s)$ and for the volatility $\sigma(P) = \sigma(x_1, \ldots, x_s)$ of the portfolio P in the time period T the following representations:

$$\mu(P) = x_1\mu_1 + x_2\mu_2 + \cdots + x_s\mu_s \tag{1}$$

$$\sigma(P) = \sqrt{\sum_{i,j=1}^{s} x_i x_j \sigma_i \sigma_j \rho_{ij}} \tag{2}$$

The opportunity set (OS) of the optimization problem is defined as

$$\{(\sigma(x_1, \ldots, x_s), \mu(x_1, \ldots, x_s)) \mid x_1 + x_2 + \cdots + x_s = 1, x_i \ge 0, \forall i\}. \tag{3}$$

If we illustrate this OS in a volatility/return diagram then typically we get sets of the form shown in Fig. 1.

If we add one further asset A_0 which is "riskless", i.e. $\sigma_0 = 0$, then the new opportunity set has the typical shape as in Fig. 2. Here the left upper boundary of the new OS is built by the tangent from A_0 to the tangent point T to the left upper boundary of the original OS and the remaining part of the left upper boundary of the original OS.

The "left upper bound" of an OS is called its "Efficient Border (EB)". It is obvious that (theoretically) only portfolios which are situated on the EB are of interest: for

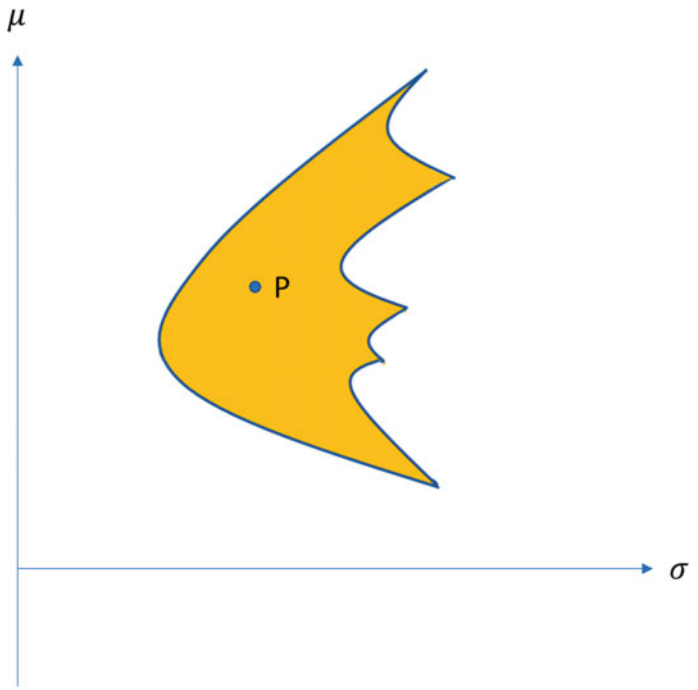


Fig. 1 Typical shape of an opportunity set without riskless asset

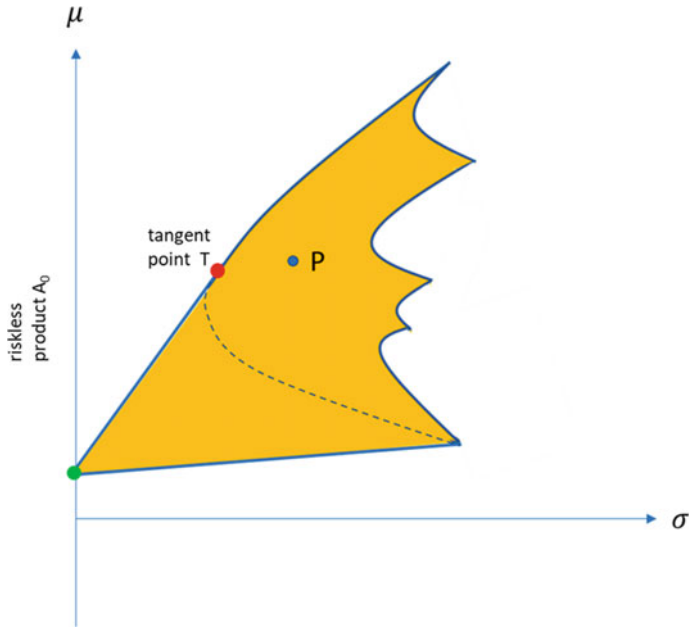


Fig. 2 Typical shape of an opportunity set with riskless asset A_0

any portfolio P in the interior of the OS there certainly exists a portfolio S on the EB such that $\sigma(S) \leq \sigma(P)$ and $\mu(S) > \mu(P)$.

The tangent point T represents the portfolio with maximal Sharpe ratio, i.e., the portfolio P for which the ratio $\frac{\mu(P)-r}{\sigma(P)}$ attains its maximal value. Here r denotes the riskfree interest rate for the time period $[0, T]$. r is also the expected return of A_0.

So the two main objectives in portfolio optimization are to determine the EB and the portfolio T (the so-called "market portfolio". This can efficiently be done with convex optimization techniques. In classical portfolio optimization, information about the concrete shape of the whole OS, i.e. especially about the interior of the OS, usually is not needed.

3 Portfolio Optimization with Many Constraints

The main problem in the application of Markowitz' portfolio optimization is to give reliable estimates for the expected returns of the assets.

In 2018 a fintech company contacted and mandated us to develop an advanced portfolio optimization tool which should improve and extend the classical approach in two aspects:

- the new technique should be less dependent on estimates for future expected returns
- it should be possible to integrate many additional constraints

The first item was managed by our team by inventing a new performance measure for assets named "fynup-ratio". This measure is based on machine learning principles and substitutes the expected return μ. Using this new measure, however, we can proceed in absolutely the same way as in classical portfolio optimization. It is not the topic of this article to explain how this new measure "fynup-ratio" is conceived and how it works, for this we refer to [6]. Hence, in the following we will just address the old concept of expected return.

The second item, however, will be in the center of the following considerations. We will first give some examples of "strict" and "soft" constraints, which should be handled with the new system.

Examples of *strict constraints* could be:

- We just consider portfolios P which contain at most k of the s assets
- Every asset in a portfolio P must appear in the portfolio with at least (or at most) $z\%$
- Assuming that each of the assets A_i (which for example may be large investment funds) has an *asset classification*, i.e., it is classified with respect to asset classes (e.g.: A_i consists of 20% stocks, 30% bonds, 10% alternative investments, 5% commodities, …). Then, (for example) every portfolio P must contain at least $a\%$ stocks, $b\%$ bonds, and $c\%$ commodities

- If we assume that each of the assets has a *sustainability parameter*, i.e., has a classification number τ between 0% and 100%, which classifies increasing sustainability, then (for example) every portfolio P must have an average sustainability of at least $y\%$.
- Assuming that each of the assets A_i has a *regional classification*, (e.g.: A_i consists of 20% US assets, 30% European assets, 10% Asian assets,…), then (for example) every portfolio P must contain at least $a\%$ US assets, and at least $b\%$ Asian assets.
- Assuming that each of the assets A_i has a *classification with respect to industries* (e.g.: A_i consists of 20% IT assets, 30% telecommunication assets, 10% chemistry assets, …). Then (for example) every portfolio P must contain at least $a\%$ IT assets, $b\%$ chemistry assets, and $c\%$ financial industry assets.

Examples of *soft constraints* could be:

– We predefine an asset class distribution (e.g.: 40% stocks, 30% bonds, 10% commodities, 20% alternative investments), then the portfolios in consideration should have an asset class distribution "as close as possible" to our desired distribution.
– We predefine a regional distribution (e.g.: 40% US, 30% Europe, 10% Asia, 10% Southern America, 10% others), then the portfolios in consideration should have a regional distribution "as close as possible" to our desired distribution.
– We pre-define a distribution with respect to industries (e.g.: 20% IT, 30% telecommunication, 10% chemistry, 10% transport, 30% others), then the portfolios in consideration should have a distribution with respect to industries "as close as possible" to our desired distribution.

Now it is our task to search for portfolios P which satisfy the strict constraints, which satisfy the soft constraints approximately, and which show a high expected return (high fynup-ratio) combined with a low volatility, i.e. with a high Sharpe ratio.

In this article we will not address this concrete challenge in detail, but we will project it in a first step to a more "abstract" problem: If we deal with constraints like above, then in general we have to consider also portfolios in the interior of the opportunity set. That means, we need also information about portfolios in the interior of the OS, hence, also on the shape of the OS when the constraints are applied. So an immediate idea would be, to approximate the OS as well as possible with the help of Monte Carlo simulation and to work with the simulated portfolios.

4 Approximation of the Opportunity Set by Naïve Monte Carlo, and by Exponential Monte Carlo

A naïve approach to the task of approximating the OS of course would be the following:

We generate a large number (say N) of random portfolios $P1^{(1)}$, $P^{(2)}, \ldots, P^{(N)}$ by generating $N \cdot s$ independent uniformly distributed pseudo-random numbers $y_i^{(j)}$ in the interval [0, 1] for $j = 1, 2, \ldots, N$ and $i = 1, 2, \ldots, s$. We set

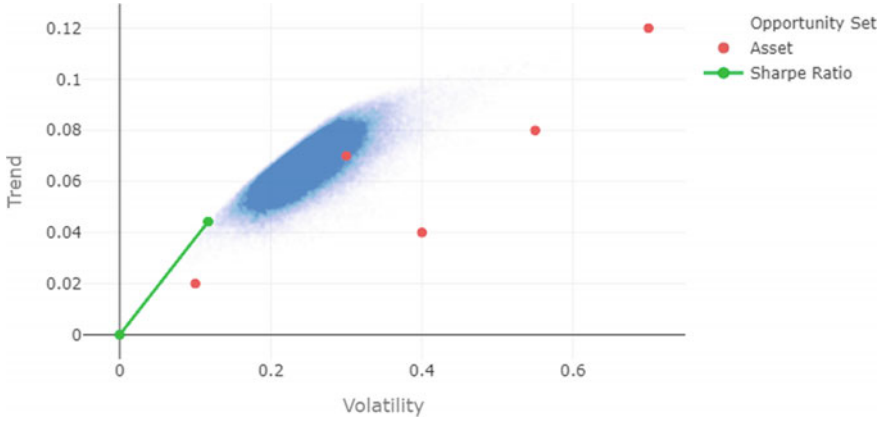


Fig. 3 Approximation of an OS with naïve MC and 150.000 sample points. The points are plotted semi-transparent to illustrate where most simulations are located

$$x_i^{(j)} := \frac{y_i^{(j)}}{y_1^{(j)} + y_2^{(j)} + \cdots + y_s^{(j)}}.$$

Then the portfolio $P^{(i)}$ is given by

$$P^{(i)} = x_1^{(i)} A_1 + x_2^{(i)} A_2 + \cdots + x_s^{(i)} A_s.$$

In Fig. 3 we see an example for an approximation of the OS of $s = 5$ assets, which are highlighted in red. This approximation was generated with the above described naïve method and with 150.000 sample weights.

We see a picture that is not satisfying. Trivially, the assets are elements of the connected (!) OS. We see a small region which is very well and densely covered by sample points, however, there are large regions (especially near some of the assets) in which we can hardly find any sample points.

One reason for this unsatisfactory situation of course is the following: The s-dimensional sample weight vectors $(x^{(1)}, x^{(2)}, \ldots, x^{(s)})$ lie on the $(s-1)$-dimensional simplex

$$H : x^{(1)} + x^{(2)} + \cdots + x^{(s)} = 1, x^{(i)} \geq 0.$$

If they are constructed with the help of uniformly distributed $y^{(i)}$ as described above, then the $(x^{(1)}, x^{(2)}, \ldots, x^{(s)})$ are not uniformly distributed in H, but they have a higher density in the center of H than close to the boundary of H. This effect can already be seen in the case $s = 3$ (see Fig. 4), but it becomes worse and worse in higher dimensions.

In the following we show how to alternatively and correctly construct uniformly distributed point sets on this simplex. Nevertheless, we have seen the above naïve

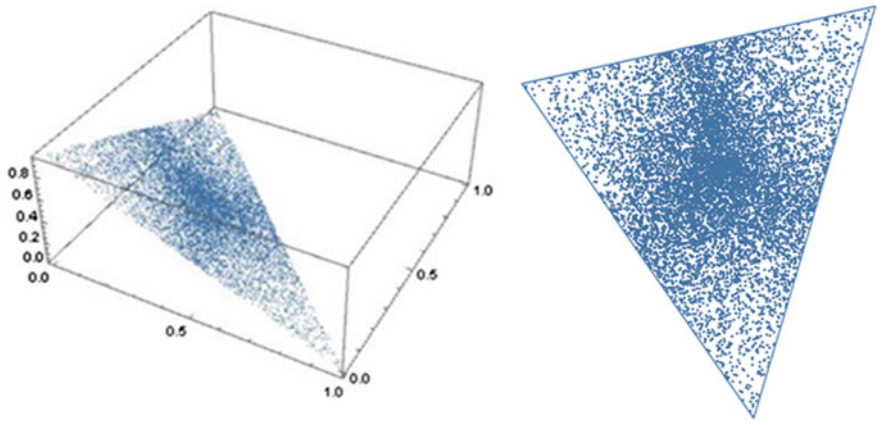


Fig. 4 Normed 3-dimensional sample weights, naïve approach, 10.000 samples

approach several times in applications. The use of this approach is not completely "wrong", since in our case we are not evaluating an integral or expected value, but we are searching for a maximum and for a dense approximation of a set. Therefore, we also give the results for the naïve approach in the following.

Certainly, the better approach is to generate the sample weights $(x^{(1)}, x^{(2)}, \ldots, x^{(s)})$ on H in such a way that we generate a "typical" distribution of $(s-1)$ points $z^{(1)}, z^{(2)}, \ldots, z^{(s-1)}$ in $[0, 1]$, and choose the $(x^{(1)}, x^{(2)}, \ldots, x^{(s)})$ as the distances between successive elements from 0, 1, and $z^{(1)}, z^{(2)}, \ldots, z^{(s-1)}$.

Distances between successive elements from a random sample are exponentially distributed. So we can generate s exponentially distributed random variables $y^{(1)}, y^{(2)}, \ldots, y^{(s)}$. Finally, we again set

$$x_i^{(j)} := \frac{y_i^{(j)}}{y_1^{(j)} + y_2^{(j)} + \ldots + y_s^{(j)}}. \tag{4}$$

Because of this normalization procedure, it does not matter which parameter λ we choose for the generation of $\mathrm{Exp}(\lambda)$ distributed $y^{(i)}$. If we proceed in this way, the distribution of Fig. 4 changes to the one illustrated in Fig. 5. In fact, we obtain uniformly distributed point sets on the $(s-1)$-dimensional simplex in this way (see [5] or [12] for more information).

If we use this exponential approach to generate samples for the approximation of the opportunity set of a portfolio optimization problem, then with the same parameters as in the example leading to Fig. 3 we now get a result as it is shown in Fig. 6. The visual impression of the new result is already considerably better than the result shown beforehand.

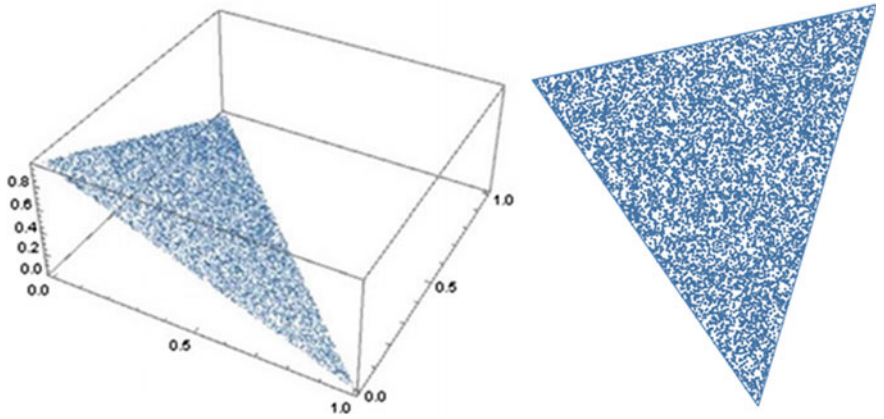


Fig. 5 Normed 3-dimensional sample-weights, exponential approach, 10.000 samples

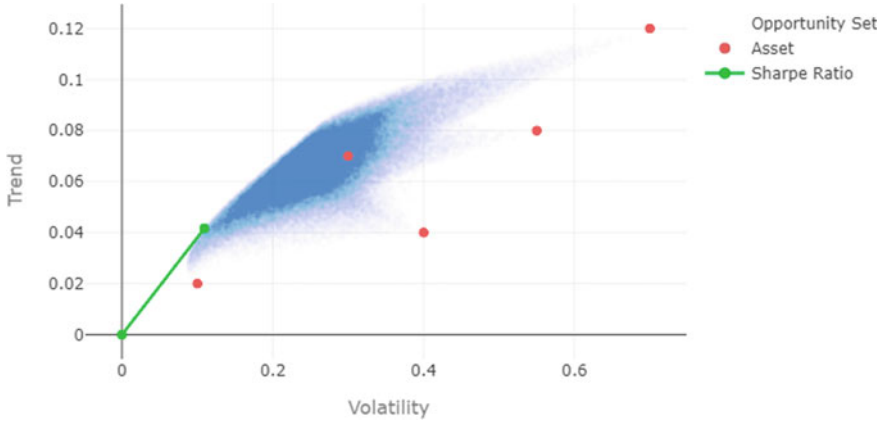


Fig. 6 Approximation of an OS with exponential MC and 150.000 sample points

5 Approximation of the Opportunity Set with Exponential QMC

When we apply MC methods for simulation, then we generate samples with the help of pseudo-random generators, i.e., with the help of pseudo-random point sets. However, for some types of problems it is advantageous to use so-called low-discrepancy (i.e. in some sense very well distributed) point sets instead of pseudo-random point sets. Examples of such low-discrepancy sequences (we will call them "QMC point sets") are Faure point sets, Sobol' point sets, Halton point sets, Niederreiter nets, good lattice point sets, …

The construction of such point sets is based on deep results and principles from various fields of mathematics, as for example number theory, combinatorics, algebraic geometry, complexity theory, among others.

If we use QMC point sets instead of pseudo-random point sets, we say that we apply Quasi-Monte Carlo methods (QMC methods). We will not go into detail on how to construct QMC point sets. There is vast excellent literature on this topic (see e.g. [9], or [4]), and there exists excellent open source software repositories where you can easily and efficiently download and use QMC point sets, for example Pierre L'Ecuyer's "LatticeFinder" on [7] (https://github.com/mungerd/latbuilder), or also the software on our homepage www.lsqf.org.

Again, just to give a first (!) visual impression for the difference between the qualities of an MC and a QMC approach to our portfolio optimization problem, we show the right hand part of Fig. 5 (projection of three-dimensional weights generated with exponential approach and a pseudo-random point-set) in comparison with the result when using a Niederreiter point-set. Here we use just 1.000 sample points, then the difference is easier to see with the bare eye. We see a more regular distribution of sample weights if we use QMC instead of MC (Fig. 7).

Of course, when using uniformly distributed (!) QMC point sets, in a first step we transform these uniformly distributed point sets by the inversion method to exponentially distributed point sets. We will call this the "exponential QMC-approach" in the following.

In Fig. 8 we show the analogue to Figs. 3 and 6 again now using Niederreiter point sets instead of pseudo-random point sets. Again we recognize a (slight, visual) improvement compared to Fig. 6.

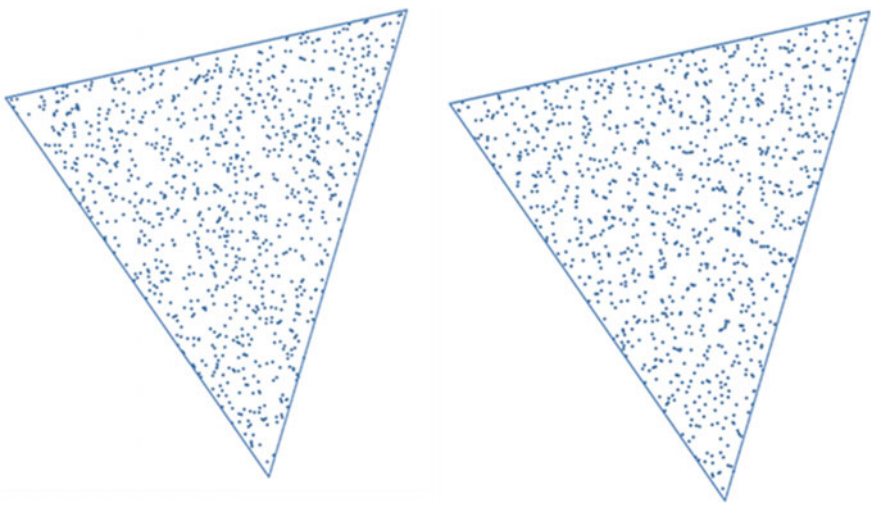


Fig. 7 Normed 3-dimensional sample-weights, exponential approach, 1.000 samples, MC left, QMC right

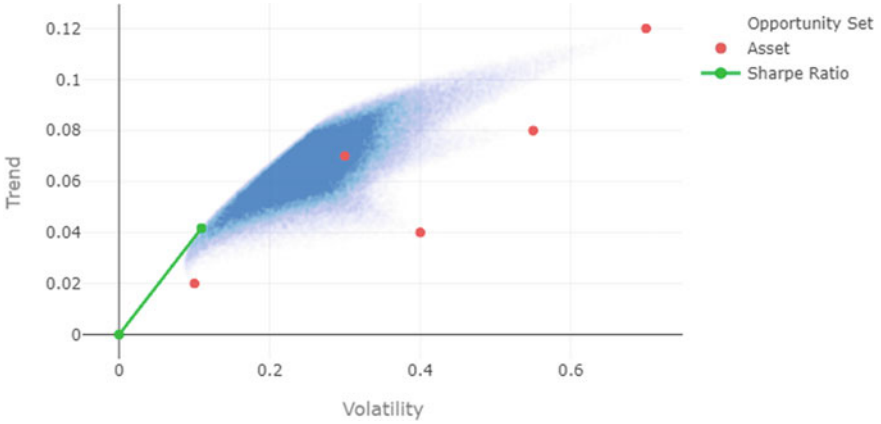


Fig. 8 Approximation of an OS with exponential Niederreiter point sets and 150.000 sample points

6 Approximating the Market Portfolio with MC, Exponential MC, and Exponential QMC

In the previous sections we always just argued with "visual superiority" of one method over the other. Of course, we also want to measure the quality of the different methods. A quick first test could be to compare the performance of the "best" portfolio given by the different approaches, which translates to finding portfolios with as high as possible Sharpe ratio (without further constraints).

We have carried out a multitude of corresponding experiments for examples with many different parameter choices. We just show three different typical examples of our results in the following three pictures.

Example 1: 5 assets. The best attained Sharpe ratios after 10.000.000 generated samples are shown in Table 1.

In Fig. 9 we illustrate the corresponding speed of convergence. While all the exponential distribution approaches are approximating the best value for Sharpe ratio in a similar speed and accuracy, the pure MC approach is noticeable slower in

Table 1 Example 1: best attained Sharpe ratios after 10.000.000 generated samples

Approach	Best simulated Sharpe ratio
MC	0.38112
Exp. MC	0.38127
Exp. Sobol'	0.38132
Exp. Niederreiter	0.38134
Exp. Faure	0.38125

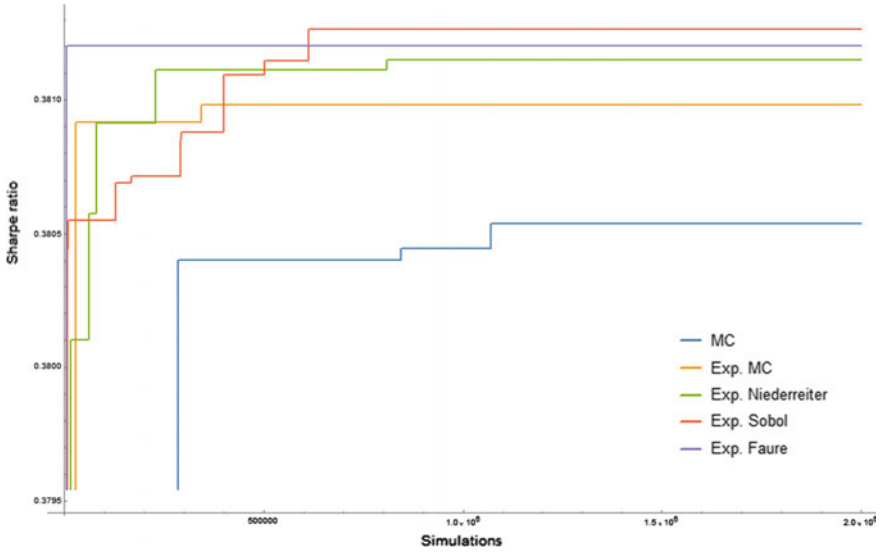


Fig. 9 Example 1: maximization of Sharpe ratio with different approaches, 5 assets

Table 2 Example 2: best attained Sharpe ratios after 10.000.000 generated samples

Approach	Best simulated Sharpe ratio
MC	1.61989
Exp. MC	2.08056
Exp. Sobol'	2.21090
Exp. Niederreiter	2.32745
Exp. Faure	2.11156

terms of number of steps needed. This phenomenon occurs not only occasionally for a single MC experiment but it consistently occurs also in repetitions of the experiment with other pseudo-random strings.

Further we see, that *for this type of problem* using QMC instead of MC hardly does provide an advantage.

Example 2: 20 assets. The best attained Sharpe ratios after 10.000.000 generated samples are shown in Table 2.

Also in the second example we recognize the superiority of the exponential distribution approach over the pure MC approach in terms of finding an approximation to the portfolio with best Sharpe ratio. However, there was not such a discrepancy in the number of steps needed to get to similarly good solutions as can be seen in Fig. 10. This behavior again could be observed for multiple simulations of the MC approach, so it is not an event that occurs for only a single MC experiment. Further,

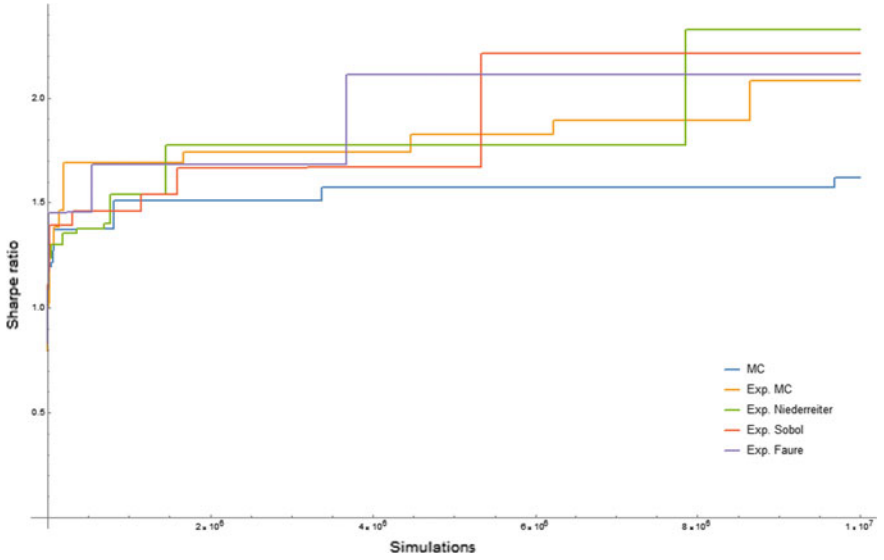


Fig. 10 Example 2: maximization of Sharpe ratio with different approaches, 20 assets

Table 3 Example 3: best attained Sharpe ratios after 10.000.000 generated samples

Approach	Best simulated Sharpe ratio
MC	1.16872
Exp. MC	1.22833
Exp. Sobol'	1.27687
Exp. Niederreiter	1.31698
Exp. Faure	1.22367

the switch from exponential MC to exponential QMC leads to slight improvements, especially for the Sobol'- and Niederreiter point sets.

Example 3: 50 assets. In the next example we consider even more assets, and look if we can again recognize an improvement of the exponential QMC methods over the exponential MC method. We see the results in Table 3 and again the development of the best attained Sharpe ratio in Fig. 11.

Summarized, we see – as expected – clearly better results, when switching from pure MC simulation to the exponential approaches. Again, the switch from exponential MC to exponential QMC methods leads to better results, the improvements are especially visible for a larger number of assets.

The investigation of the maximization of the Sharpe ratio was just a first test for the effectiveness of the different methods. (Of course this task can be carried out directly or with adaptive methods much more successfully than with MC- and

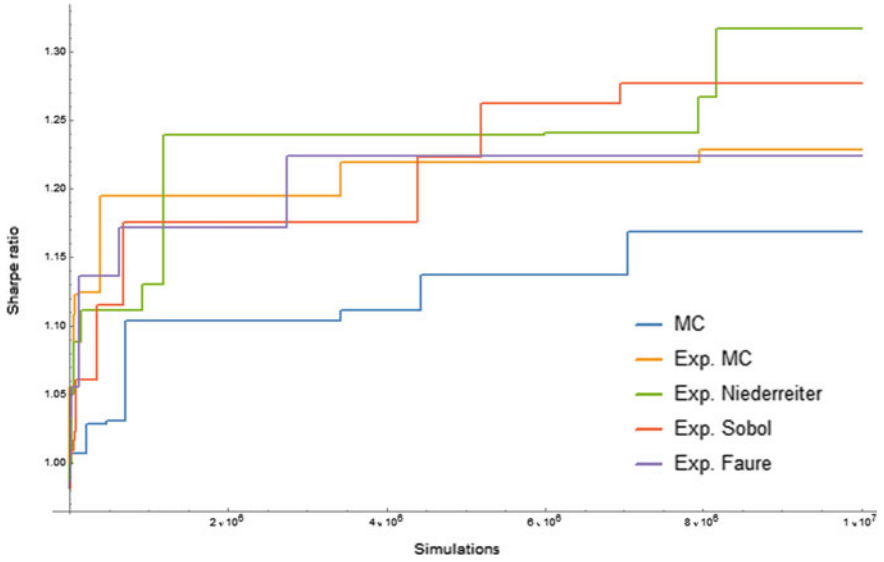


Fig. 11 Example 3: maximization of Sharpe ratio with different approaches, 50 assets

QMC-simulation!) The definite goal in fact is to carry out portfolio optimization with many constraints—as explained in Sect. 3—in an efficient way.

Recall, we searched for example for portfolios with, e.g., a very high sustainability, with strong focus on Asian technology in form of alternative investment assets, with a Sharpe ratio as high as possible, ….

Thus, we *need to approximate the whole opportunity set most efficiently* and not only the region of the portfolio with the global maximum in terms of Sharpe ratio.

7 Approximating the Whole OS with MC, Exponential MC, and Exponential QMC

First we have to clarify what we mean by a "good approximation of the OS". If we carry out an approximation of the OS, with the help of N sample portfolios, then of course we do not want to have large "empty regions" in the OS, which do not contain a sample portfolio. A possible suitable measure would be the "dispersion" of the sample set in the OS.

If we denote with $P_1, P_2, \ldots, P_N$ the position of the N sample points in the opportunity set OS and if we denote with *dist* the distance given by the max-norm, then we would like to calculate

$$disp_{OS}(P_1, P_2, \ldots, P_N) := \sup_{x \in OS} \min_{i=1,2,\ldots,N} dist(x, P_i) \tag{5}$$

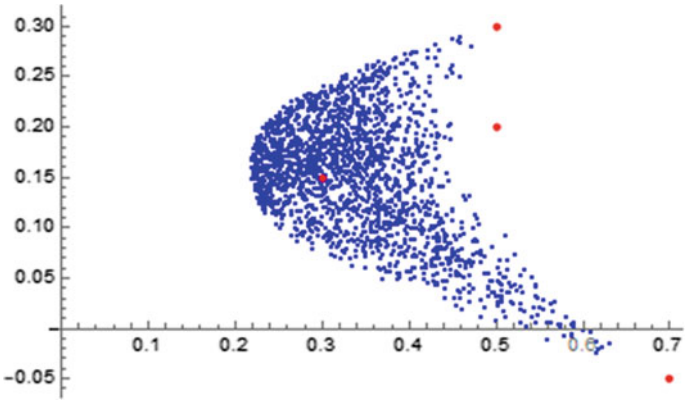


Fig. 12 Calculate the dispersion of this 1,000-point sample set in its OS!

That means, we would like to calculate half the length of the sides of the largest square with center in the opportunity set, which contains none of the sample points. We call $disp_{OS}(P_1, P_2, \ldots, P_N)$ the dispersion of the sample set in the opportunity set. Of course we would like to have sample sets with dispersion as small as possible. However, if we want to calculate for example the dispersion of the sample set consisting of 1,000 samples shown in Fig. 12, then there arise two main problems.

1. Find an efficient algorithm to calculate the dispersion at least approximately
2. We do not know the exact shape of the opportunity set in advance, so, how is one able to calculate the $\sup_{x \in OS}$ in the definition of the dispersion?

Especially the second problem is a severe restriction in this regard.

8 How to Calculate the Dispersion of a Sample Set in an OS?

For our experimental environment, in which we principally test the general approximation properties of exponential QMC for opportunity sets compared to naïve and exponential MC, we proceed in the following way:

- For each of our given parameter sets for the portfolio problems, we first approximate the opportunity set with exponential QMC with M simulations, where M is a number of sample points which is much larger, than the number of simulations N of the sample points which are tested with our experiments.
- In Fig. 13 we see the 1,000 sample points $P_1, P_2, \ldots, P_N$ we want to test, and which are shown already in Fig. 12 in blue, and in yellow we have approximated the OS with the help of 10,000,000 sample points, $Q_1, Q_2, \ldots, Q_M$.

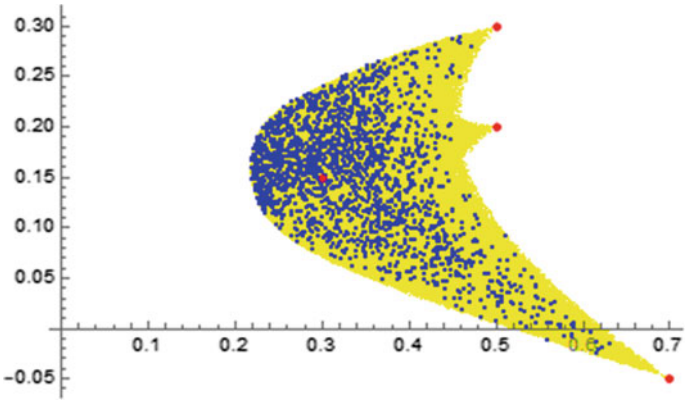


Fig. 13 Calculate the dispersion of the blue 1,000-point sample set in the yellow approximation of the OS!

Then we approximate the dispersion $disp_{OS}$ from Eq. (5) by the following quantity:

$$\max_{j=1,2,\ldots,M} \min_{i=1,2,\ldots,N} dist(Q_j, P_i) \tag{6}$$

There remains the question on how to efficiently calculate $min_{i=1,2,\ldots,N} dist(Q_j, P_i)$.

This task could simply be accomplished by calculating all distances of Q_j to each of the points in set $\{P_i\}_{i=1,\ldots,N}$, however, this is impractical due to the enormous amount of distances, which would have to be determined. Much more efficient is the following method:

1. We make sure the test set $\{P_i\}_{i=1,\ldots,N}$, with each point being $(vola(P_i), trend(P_i))$ is ordered by one of the two axis. We chose ordering by the volatility, and assume this type of ordering in the further steps.
2. For each point Q_j we execute the following:
   a. Determine the closest point in terms of the *vola*-axis in $\{P_i\}_{i=1,\ldots,N}$ (which can be done in $\mathcal{O}(\log N)$ steps with a modified binary search algorithm, due to the previously executed sorting). We denote this closest point with P_i.
   b. Starting from P_i, we calculate the distance to Q_j for the points P_i, P_{i+1}, $P_{i+2}, \ldots$ and always remember the determined minimal distance to Q_j, until the distance on the *vola*-axis is already greater than the current minimal distance.
   c. We carry out the same procedure "downwards", i.e. for P_{i-1}, $P_{i-2}, \ldots$ again until the distance on the *vola*-axis is already greater than the current minimal distance.

By repeating this for every point in $\{Q_j\}_{j=1,\ldots,M}$ we finally get the desired value for Eq. (6). By proceeding in the described way we finally get the overall minimal

distance with much less steps – and therefore time – taken. In fact, in our examples of Sect. 9, where we used up to $N = 10^5$ and $M = 10^7$, the above algorithm only took around 10 seconds for the search, while the brute force would have needed around 4 hours on our machine. And this only came with the cost of sorting the set once.

9 Some Simulation Results

In the following we give some selected examples for the typical outcomes in our simulations and the corresponding analysis of the quality of the approximations of opportunity sets with different approaches. As before, because of the random behavior in this type of simulations all the results were again rechecked in multiple simulations to ensure these numbers were no outliers.

In all the following examples we used a set with 10,000,000 samples, produced by the exponential QMC (Niederreiter) method, as our approximation for the opportunity set, denoted by $\{Q_j\}$ in the description in Sect. 8.

Example 1: 5 assets (as in Fig. 8). In Table 4 we see the dispersion in the point sets generated by the different approaches. Whilst we saw not much improvement when using exp. QMC over exp. MC in example 1 of Sect. 6, we see a clear improvement in terms of dispersion already in this example.

One remarkable fact (but not entirely surprising when we look at Fig. 8) is, that the largest dispersion always occurs in the same area in our opportunity set. The sets are always dispersed the most at the upper right corner, i.e. really close to the asset at the top right of our vola-trend chart. The fact that the greatest dispersion is close to the single assets becomes even more pronounced when the number of assets is increased, as we see in further examples.

To also get a feeling about the dispersion in the denser part of the opportunity set we executed another test, where we only checked the dispersion on $\{Q_j\}$ for portfolios with trend above 5% and volatility below 30%. This is especially interesting for us, since restrictions of these types would also be a standard requirement of customers when looking for suitable portfolios. The (again) promising results can

Table 4 Example 1: dispersion after 40,000 respectively 100,000 generated samples in our test sets

Approach	Dispersion (40,000 samples)	Dispersion (100,000 samples)
MC	0.11993	0.10159
Exp. MC	0.04452	0.03368
Exp. Sobol'	**0.02357**	**0.01603**
Exp. Niederreiter	0.03207	0.02235
Exp. Faure	0.02686	0.01721

Table 5 Example 1: dispersion after 40,000 respectively 100,000 generated samples in our test sets with respect to the opportunity set with $\mu \geq 0.05$ and $\sigma \leq 0.3$

Approach	Dispersion (40,000 samples)	Dispersion (100,000 samples)
MC	0.0024311	0.0015621
Exp. MC	0.0016736	0.0013923
Exp. Sobol'	**0.0014320**	**0.0012667**
Exp. Niederreiter	0.0014699	0.0012971
Exp. Faure	0.0016503	0.0013572

Table 6 Example 2: dispersion after 40,000 respectively 100,000 generated samples in our test sets

Approach	Dispersion (40,000 samples)	Dispersion (100,000 samples)
MC	0.320021	0.315001
Exp. MC	0.118638	0.108512
Exp. Sobol'	**0.107355**	**0.075218**
Exp. Niederreiter	0.103512	0.097131
Exp. Faure	0.108361	0.077263

be found in Table 5.

Example 2: 20 assets. In example 2 we are again interested in the behavior of the dispersion property, when the number of assets increases. Table 6 again indicates strong improvement when switching to exponential approaches, and using QMC again gives slightly better results.

However, we have to highlight some strong caveat: Even when using exponential QMC methods and lots of sample points the approximation of the actual opportunity set is not really satisfying, as can be seen in Fig. 14. Diversification effects lead to an accumulation of the portfolios in a certain area of the opportunity set, whilst other areas are barely covered. We are going to address this problem again in Sect. 10. Because of this behavior Table 7 is more reliable than the results from Table 6. Here, we used restrictions on the trend and volatility as before, with trend at least at 4% and maximal volatility at 20%.

10 Conclusions, Outlook, and Further Practical Problem

The above (and many more) examples have convinced us, that it doubtlessly makes sense to integrate exponential QMC methods in our commercial software project with the goal to carry out portfolio optimization with many constraints. Exponential QMC never gave worse results than exponential MC and in many cases led to recognizable

Table 7 Example 2: dispersion after 40,000 respectively 100,000 generated samples in our test sets with respect to the opportunity set with $\mu \geq 0.04$ and $\sigma \leq 0.2$

Approach	Dispersion (40,000 samples)	Dispersion (100,000 samples)
MC	0.0216138	0.0179415
Exp. MC	0.0164915	0.0096525
Exp. Sobol'	**0.0161258**	**0.0081269**
Exp. Niederreiter	0.0123436	0.0098834
Exp. Faure	0.0186645	0.0115707

improvements (especially Niederreiter sequences seemed to work very well on a consistent basis). However, several practical problems still have to be managed. For example, we will also have to compare our advanced approach and its potential superiority to other approaches reliably in the much more complex real-world setting and not only in the artificially simplified environment, which we have described and analyzed above.

Also the problem of worse coverage of the opportunity set when looking at a larger number of assets (illustrated in Fig. 14), needs to be taken care of. One idea to tackle this problem would be to use a certain fraction of our samples for portfolios restricted on subsets of our assets. I.e. if we simulate, say 10,000,000 samples of our portfolios, we could use one half or two thirds of these simulations on portfolios of size 5 instead of 20. The subsets would again be chosen randomly. We assume this procedure would cover the opportunity set as a whole better, than just using portfolios, which include all assets. The details like, how to choose these subsets or their sizes in a smart way, are still open for research.

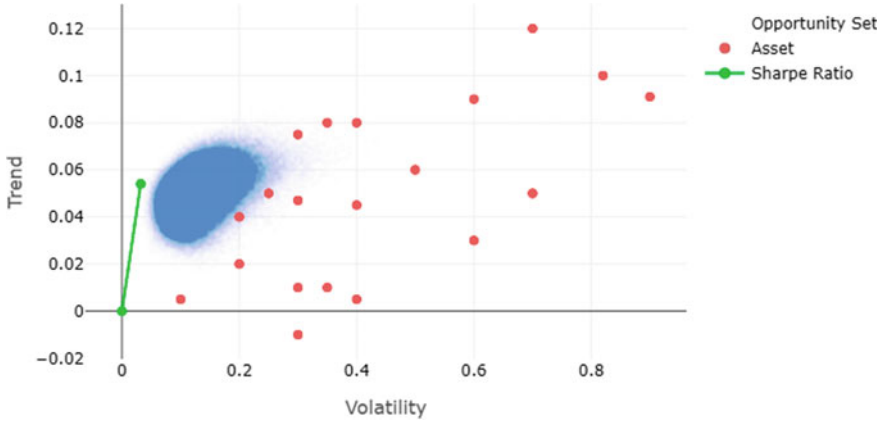


Fig. 14 Approximation of an OS with exponential QMC (Niederreiter) and 500.000 sample points

Additionally, this issue is closely connected to one further practical problem, we are faced with: A further constraint that can be given in a portfolio optimization problem (and it was indeed requested by our industry-partner that these constraints also can be handled) is the following:

We have a rather large universe of, say, $S = 50$ possible assets. However, the portfolios which we generate, always must contain at most, say, $k = 10$ of these assets. How should we proceed in this case, to approximate the corresponding opportunity set in a best possible way?

Of course we can proceed with a naïve MC approach: In a first step we choose a random-k-tuple out of the S assets. In a second step we generate a k-tuple of weights for this choice of assets. Probably, in this situation we will not benefit from choosing exponentially distributed weights. Note, that

$$\binom{50}{10} \cong 10^{10},$$

hence, in a simulation experiment any chosen k-tuple of assets will appear at most once. So, it does not really make sense to choose the only relevant weight sequence for this k-tuple exponentially.

Thus, the question is: Is it possible to improve the described naïve MC approach in this situation in any way by some exponential distribution approach and/or by applying QMC instead of MC?

Acknowledgements The authors are supported by the Austrian Science Fund (FWF), Project F5507-N26, which is part of the Special Research Program Quasi-Monte Carlo Methods: Theory and Applications, and by the Land Upper Austria research funding. The authors thank an anonymous referee for very carefully reading our manuscript and for many valuable remarks and suggestions for improving this paper!

References

1. Boyle, P., Imai, J., Tan, K.: Computation of optimal portfolios using simulation-based dimension reduction. Insur. Math. Econ. **43**, 327–338 (2008). https://doi.org/10.1016/j.insmatheco.2008.05.004
2. Cvitanić, J., Goukasian, L., Zapatero, F.: Monte Carlo computation of optimal portfolios in complete markets. J. Econ. Dyn. Control **27**(6), 971–986 (2003). https://doi.org/10.1016/S0165-1889(02)00051-9. https://www.sciencedirect.com/science/article/pii/S0165188902000519. High-Performance Computing for Financial Planning
3. Detemple, J.B., Garcia, R., Rindisbacher, M.: A Monte Carlo method for optimal portfolios. J. Financ. **58**(1), 401–446 (2003). http://www.jstor.org/stable/3094492
4. Dick, J.: Digital Nets and Sequences: Discrepancy Theory and Quasi-monte Carlo Integration, 1st edn. Cambridge Univ. Press (2010). http://media.obvsg.at/AC08318702-1001
5. Grimme, C.: Picking a uniformly random point from an arbitrary simplex (2015). https://doi.org/10.13140/RG.2.1.3807.6968
6. Larcher, G.: Quantitative Finance: Strategien, Investments, Analysen, 1st edn. Springer Fachmedien Wiesbaden GmbH and Springer Gabler, Wiesbaden (2020)

7. L'Ecuyer, P.: Latnet builder — a general software tool for constructing highly uniform point sets. https://github.com/umontreal-simul/latnetbuilder. Accessed: 2021-10-27
8. Markowitz, H.: Portfolio selection. J. Financ **7**(1), 77–91 (1952). http://www.jstor.org/stable/2975974
9. Niederreiter, H.: Random number generation and quasi-Monte Carlo methods. (AT-OBV)AC00146588 63. Society for Industrial and Applied Mathematics (1992). http://media.obvsg.at/AC00534310-1001
10. Perrin, S., Roncalli, T.: Machine learning optimization algorithms & portfolio allocation. SSRN Electron. J. (2019). https://doi.org/10.2139/ssrn.3425827
11. Rometsch, M.: Quasi-Monte Carlo methods in finance: with application to optimal asset allocation (2008). https://www.wiso-net.de/document/DIPL,ADIPL__9783836616645126
12. Smith, N.A., Tromble, R.W.: Sampling uniformly from the unit simplex. Technical report, Johns Hopkins University (2004)

Geometric-Moment Contraction of G/G/1 Waiting Times

Kemal Dinçer Dingeç, Christos Alexopoulos, David Goldsman, Athanasios Lolos, and James R. Wilson

Abstract For asymptotically valid point and confidence-interval (CI) estimation of steady-state quantiles in dependent simulation output processes, two recent output-analysis procedures assume that those processes satisfy the geometric-moment contraction (GMC) condition. Moreover, the GMC condition ensures satisfaction of most of the other assumptions underlying those procedures, which are based on the techniques of batch means and standardized time series, respectively. For performance evaluation of the associated point and CI estimators, the G/G/1 queueing system provides gold-standard test processes. We prove that the GMC condition holds for G/G/1 queue-waiting times obtained with a non-heavy-tailed service-time distribution (i.e., its moment generating function exists in a neighborhood of zero). This result complements earlier proofs that the GMC condition holds for many widely used time-series and Markov-chain processes. A robustness study illustrates empirical verification of the GMC condition for M/G/1 queue-waiting times obtained with non-heavy-tailed and heavy-tailed service-time distributions.

Keywords Dependent simulation output processes · Geometric-moment contraction condition · Queue-waiting times · Single-server queue · Steady-state quantile estimation

K. D. Dingeç
Gebze Technical University, 41400 Gebze, Kocaeli, Turkey
e-mail: kdingec@yahoo.com; kdingec@gtu.edu.tr

C. Alexopoulos · D. Goldsman · A. Lolos
Georgia Institute of Technology, H. Milton Stewart School of Industrial and Systems Engineering, Atlanta, GA 30332-0205, USA
e-mail: christos@gatech.edu

D. Goldsman
e-mail: sman@gatech.edu

A. Lolos
e-mail: thnlolos@gatech.edu

J. R. Wilson (✉)
Edward P. Fitts Department of Industrial and Systems Engineering, North Carolina State University, Raleigh, NC 27695-7906, USA
e-mail: jwilson@ncsu.edu

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_6

1 Introduction

Simulation output analysis has been an active area of research for many years, particularly in the context of point and confidence-interval (CI) estimation for quantities such as the steady-state mean and quantiles of a simulation-generated process. This analysis is difficult to carry out because the observed simulation output is often non-stationary, serially correlated, and non-normal—as typified, for example, by the sequence of consecutive queue-waiting times for customers in a single-server queueing system that has the empty-and-idle initial condition and a high long-run server utilization. Various well-known output analysis methodologies (e.g., batch means, standardized time series, etc.) assume sufficient, but difficult-to-check, moment and mixing conditions for those methods to work as advertised. In this article, we consider a different underlying assumption that facilitates the development of asymptotically valid point and CI estimators of the mean or quantiles of the steady-state output-response distribution—namely, the geometric-moment contraction (GMC) condition (cf. [25]). We begin by defining the GMC condition.

Definition 1 Consider a stochastic process $\{\mathfrak{X}_k : k \geq 0\}$ that is defined by a function $\xi(\cdot)$ of a sequence of independent and identically distributed (i.i.d.) random variables (r.v.'s) $\{\varepsilon_j : j \in \mathbb{Z}\}$ such that $\mathfrak{X}_k = \xi(\ldots, \varepsilon_{k-1}, \varepsilon_k)$ for $k \geq 0$. We say that $\{\mathfrak{X}_k : k \geq 0\}$ satisfies the *GMC condition* if there exist constants $\psi > 0, C > 0$, and $r \in (0, 1)$ such that for two independent sequences $\{\varepsilon_j : j \in \mathbb{Z}\}$ and $\{\varepsilon_j^* : j \in \mathbb{Z}\}$ each consisting of i.i.d. r.v.'s distributed like ε_0, we have

$$\mathrm{E}\left[\left|\xi(\ldots, \varepsilon_{-1}, \varepsilon_0, \varepsilon_1, \ldots, \varepsilon_k) - \xi\left(\ldots, \varepsilon_{-1}^*, \varepsilon_0^*, \varepsilon_1, \ldots, \varepsilon_k\right)\right|^{\psi}\right] \leq C r^k \quad \text{for } k \geq 0. \tag{1}$$

The setting for the GMC condition (1) is that we have two replications of the simulation with the response function $\xi(\cdot)$ that are driven by two independent streams of random numbers as specified by Eq. (1) so that (a) the runs are initialized independently in steady-state operation at time 0, perhaps using preliminary warm-up periods for each run that are respectively based on the independent streams of random numbers $\{\ldots, \varepsilon_{-1}, \varepsilon_0\}$ and $\{\ldots, \varepsilon_{-1}^*, \varepsilon_0^*\}$; and (b) subsequently the runs share the common random numbers $\{\varepsilon_1, \ldots, \varepsilon_k\}$ from time 1 to the current time $k \geq 1$. The GMC condition requires that the difference between the paired output responses generated by the two simulations at time k will converge to zero in the mean of order ψ as the time index $k \to \infty$. Hence the difference between the paired responses also converges in probability to zero as $k \to \infty$.

In [25], it is argued that the GMC condition is easier to verify than a mixing condition such as α-, ρ-, or ϕ-mixing; and Remark 2 in [4] details problems with verifying the latter mixing conditions. On the other hand, the GMC condition has been proved to hold for the widely used finite-order moving average (MA), autoregressive (AR), and autoregressive–moving average (ARMA) processes; see Theorem 5.2 in [23]. The latter theorem in conjunction with Theorem 2 in [26] suffice

to show that the GMC condition is satisfied by a plethora of other linear and nonlinear processes exhibiting short- or long-range dependence, such as: autoregressive conditional heteroscedastic (ARCH) processes [14]; generalized autoregressive conditional heteroscedastic (GARCH) processes [9]; ARMA–ARCH and ARMA–GARCH processes [19]; random coefficient autoregressive (RCA) processes [21]; threshold autoregressive (TAR) processes [24]; and a large class of Markov chains [27]. For performance evaluation of output-analysis procedures, the G/G/1 queueing system provides gold-standard test processes; however, to the best of our knowledge, the GMC condition has not been proved to hold for any of those test processes. In this paper we prove that the GMC condition holds for the queue-waiting-time process in the G/G/1 queueing system with non-heavy-tailed service times (i.e., their moment generating function exists).

In the context of formulating asymptotically valid point and CI estimators of steady-state quantiles, the GMC condition plays a critical role in recently developed output-analysis procedures based on the techniques of batch means [5, 6] and standardized time series [3, 4]. Some additional notation is required to explain clearly and concisely the significance of the GMC condition in this context. For the steady-state simulation response $\mathcal{X}$ and for each $\mathfrak{x} \in \mathbb{R}$, we let $F(\mathfrak{x}) \equiv \Pr\{\mathcal{X} \le \mathfrak{x}\}$ denote the cumulative distribution function (c.d.f.) of $\mathcal{X}$. Given $p \in (0, 1)$, we seek to estimate the p-quantile of the response, $\mathfrak{x}_p \equiv F^{-1}(p) \equiv \inf\{\mathfrak{x} : F(\mathfrak{x}) \ge p\}$; and we let $f(\mathfrak{x})$ denote the probability density function (p.d.f.) of $F(\mathfrak{x})$. For each $k \ge 1$, we define the indicator r.v. $I_k(\mathfrak{x}) \equiv 1$ if $\mathfrak{x}_k \le \mathfrak{x}$, and $I_k(\mathfrak{x}) \equiv 0$ otherwise. Using the series of responses $\{\mathcal{X}_1, \ldots, \mathcal{X}_n\}$ of length $n \ge 1$, we sort the responses in ascending order to obtain the order statistics $\mathcal{X}_{(1)} \le \cdots \le \mathcal{X}_{(n)}$. The point estimator of $\mathfrak{x}_p$ is defined as $\widetilde{\mathfrak{x}}_p(n) \equiv \mathcal{X}_{(\lceil np \rceil)}$, where $\lceil \cdot \rceil$ denotes the ceiling function. For a sample of size $n \ge 1$, we let $\bar{I}(\mathfrak{x}, n) \equiv n^{-1} \sum_{k=1}^{n} I_k(\mathfrak{x})$; and for each $\ell \in \mathbb{Z}$, we let $\rho_I(\mathfrak{x}, \ell) \equiv \mathrm{Corr}[I_k(\mathfrak{x}), I_{k+\ell}(\mathfrak{x})]$ denote the autocorrelation at lag ℓ in the indicator process $\{I_k(\mathfrak{x}) : k \ge 1\}$. With this setup, we can define another key assumption of our quantile-estimation procedures.

Definition 2 The indicator process $\{I_k(\mathfrak{x}_p) : k \ge 1\}$ has the *short-range dependence (SRD) property* if

$$0 < \sum_{\ell \in \mathbb{Z}} \rho_I(\mathfrak{x}_p, \ell) \le \sum_{\ell \in \mathbb{Z}} |\rho_I(\mathfrak{x}_p, \ell)| < \infty \tag{2}$$

[8, p. 7]. If the SRD property holds, then the *variance parameters* for the r.v.'s $\bar{I}(\mathfrak{x}_p, n)$ and $\widetilde{\mathfrak{x}}_p(n)$ satisfy the relations

$$\left.\begin{aligned} \sigma_I^2 &\equiv \lim_{n \to \infty} n \mathrm{Var}\left[\bar{I}(\mathfrak{x}_p, n)\right] = p(1-p) \sum_{\ell \in \mathbb{Z}} \rho_I(\mathfrak{x}_p, \ell) \in (0, \infty), \\ \sigma^2 &\equiv \lim_{n \to \infty} n \mathrm{Var}\left[\widetilde{\mathfrak{x}}_p(n)\right] = \frac{\sigma_I^2}{f^2(\mathfrak{x}_p)} \in (0, \infty). \end{aligned}\right\} \tag{3}$$

The SRD property is assumed for all simulation output-analysis procedures based on the techniques of batch means or standardized time series; and in general this property is relatively difficult to verify either empirically or theoretically [1, 2, 10, 15]. It was proved recently in [13] that if the output process $\{\mathfrak{X}_k : k \geq 1\}$ satisfies the GMC condition, then the associated indicator process $\{I_k(\mathfrak{x}_p) : k \geq 1\}$ satisfies the SRD condition.

Another key assumption for developing asymptotically valid point and CI estimates of $\mathfrak{x}_p$ is that the indicator process $\{I_k(\mathfrak{x}_p) : k \geq 1\}$ must satisfy a certain functional limit theorem (FCLT) as detailed in [5, Eqs. (7) and (8)] and [4, Sect. 2.2.4]; moreover, as explained in [4, Remark 3], for all practical purposes verifying the SRD condition is also considered adequate verification of the FCLT condition. Thus in the context of steady-state quantile estimation, verifying the GMC condition is tantamount to verifying three of the four assumptions underlying the procedures in [4, 5].

The rest of this article is organized as follows. In Sect. 2 we present the main results, which establish that the GMC condition is satisfied for the sequence of consecutive queue-waiting times for a stable G/G/1 queueing system with a non-heavy-tailed service-time distribution. In Sect. 3 we discuss a Monte Carlo study illustrating small-sample and robustness performance related to the GMC condition for a variety of M/G/1 queueing systems with non-heavy-tailed and heavy-tailed service-time distributions. In Sect. 4 we recapitulate our conclusions, and we make recommendations for future work.

2 Main Results

This paper establishes that the GMC condition holds for the waiting-time process arising from a steady-state G/G/1 queueing system with traffic intensity less than unity and a non-heavy-tailed service-time distribution. (The term *heavy-tailed distribution* is formally defined to mean that the distribution's moment generating function (m.g.f.) does not exist in any neighborhood of the origin; thus in this article the term *non-heavy-tailed distribution* applies to a distribution whose m.g.f. does exist in a neighborhood of the origin. On the other hand, the term *light-tailed distribution* does not have a universally accepted formal definition; see [22, pp. 33–34].) Thus we consider a G/G/1 queueing system with i.i.d. interarrival times $T_0, T_1, T_2, \ldots$ having mean $\mathrm{E}[T_k] < \infty$; i.i.d., non-heavy-tailed service times $S_0, S_1, S_2, \ldots$ having mean $\mathrm{E}[S_k] < \infty$; and server utilization $\rho \equiv \mathrm{E}[S_k]/\mathrm{E}[T_k] \in (0, 1)$. In the usual parlance, T_k is the interarrival time between the k-th and $(k+1)$-st customers, and S_k is the service time of the k-th customer. For $k = 0, 1, 2, \ldots$, let W_k denote the waiting time in the queue for the k-th customer. The well-known Lindley recursion gives an easy way to calculate the waiting times,

$$W_{k+1} = [W_k + S_k - T_k]^+ \quad \text{for} \quad k = 0, 1, 2, \ldots, \tag{4}$$

where $x^+ \equiv \max\{x, 0\}$. Denote the cumulative distribution function (c.d.f.) of the steady-state waiting time by $F(x)$; this is a mixture of a point probability at zero and a c.d.f. with positive support for $x > 0$.

Example 1: For the M/M/1 queue with exponential interarrivals and services with respective rates $\lambda = 1/\mathrm{E}[T_k]$ and $\mu = 1/\mathrm{E}[S_k]$, we have

$$F(x) = \begin{cases} 0, & \text{if } x < 0, \\ 1 - \rho, & \text{if } x = 0, \\ 1 - \rho e^{-\gamma x}, & \text{if } x > 0, \end{cases}$$

where $\gamma \equiv \mu(1-\rho) = \lambda(1-\rho)/\rho$; see, e.g., p. 4 of [7]. □

We go over some additional notation and set the stage for our main Theorem 1 below. For $k = 1, 2, \ldots$, and any $v \geq 0$, let the random function $W_k(v)$ denote the waiting time of the k-th customer given that the waiting time $W_0 = v$ has occurred for the 0-th customer, whose arrival marks the start of system operation (i.e., simulation time 0). This random function is formally defined in Eqs. (6)–(8) below based on the Lindley recursion. Let V_1 and V_2 be two i.i.d. r.v.'s having c.d.f. $F(x)$, that are also independent of the S_i's and T_i's. In this case, the GMC condition (1) can be written as

$$\mathrm{E}\left[\left|W_k(V_1) - W_k^*(V_2)\right|^\psi\right] \leq Cr^k \quad \text{for } k = 0, 1, 2, \ldots, \tag{5}$$

where for all $v_1, v_2 \geq 0$, we define the recursive random functions

$$W_k(v_1) \equiv \begin{cases} v_1, & \text{if } k = 0, \\ \left[W_{k-1}(v_1) + X_{k-1}\right]^+, & \text{if } k \geq 1; \end{cases} \quad \text{and} \tag{6}$$

$$W_k^*(v_2) \equiv \begin{cases} v_2, & \text{if } k = 0, \\ \left[W_{k-1}^*(v_2) + X_{k-1}\right]^+, & \text{if } k \geq 1; \end{cases} \tag{7}$$

and the r.v.'s

$$X_i \equiv \begin{cases} 0, & \text{if } i = -1, \\ S_i - T_i, & \text{if } i = 0, 1, 2, \ldots. \end{cases} \tag{8}$$

(The definition $X_{-1} \equiv 0$ is made for notational convenience in some of the following expressions.) In this setting of the G/G/1 waiting-time process, Eqs. (6)–(8) are equivalent to using common random numbers for the interarrival and service times of customers $1, 2, \ldots$, while generating the waiting time of the "initial" customer (customer 0) from the c.d.f. $F(\cdot)$. The GMC condition (1) quantifies the intuition that any transient effect in the G/G/1 waiting-time process due to its initial condition decays geometrically fast as the customer index increases without bound.

From Eq. (2.3) of [20], we obtain the key representations

$$\left.\begin{aligned} W_k(V_1) &= W_k(0) + \big(V_1 + \min\{U_j : j = -1, 0, \ldots, k-1\}\big)^+ \\ W_k^*(V_2) &= W_k(0) + \big(V_2 + \min\{U_j : j = -1, 0, \ldots, k-1\}\big)^+ \end{aligned}\right\} \text{ for } k = 0, 1, 2, \ldots,$$

where

$$U_j \equiv \sum_{i=-1}^{j} X_i \text{ for } j = -1, 0, 1 \ldots,$$

is a random walk with the i.i.d. increments $\{X_i : i = 0, 1, \ldots\}$. Let

$$M_k \equiv \min\{U_j : j = -1, 0, \ldots, k-1\} \text{ for } k = 0, 1, 2, \ldots,$$

denote the minimum of the random walk prior to the arrival of the k-th customer. Since $U_{-1} = X_{-1} = 0$, we have $M_0 = 0$ and $M_k \le 0$ for $k = 1, 2, \ldots$; and hence M_k tends toward progressively smaller negative numbers as k grows without bound. The difference between $W_k(V_1)$ and $W_k^*(V_2)$ is given by

$$\begin{aligned} W_k(V_1) - W_k^*(V_2) &= (V_1 + M_k)^+ - (V_2 + M_k)^+ \\ &= \left\{\begin{array}{ll} 0, & \text{if } V_1 \le -M_k \text{ and } V_2 \le -M_k, \\ V_1 - V_2, & \text{if } V_1 > -M_k \text{ and } V_2 > -M_k, \\ V_1 + M_k, & \text{if } V_1 > -M_k \text{ and } V_2 \le -M_k, \\ -(V_2 + M_k), & \text{if } V_1 \le -M_k \text{ and } V_2 > -M_k \end{array}\right\} \text{ for } k = 0, 1, \ldots . \end{aligned}$$

This difference depends on V_1 and V_2, which are i.i.d. and independent of the stochastic process $\{M_k : k = 0, 1, \ldots\}$. By conditioning and the fact that V_1 and V_2 are identically distributed, we obtain, for $k = 0, 1, 2, \ldots$,

$$\begin{aligned} &\mathrm{E}\Big[\,|W_k(V_1) - W_k^*(V_2)|^\psi\Big] \\ &\quad = \mathrm{E}\Big[\,|V_1 - V_2|^\psi \mid V_1 > -M_k, V_2 > -M_k\Big] \Pr(V_1 > -M_k, V_2 > -M_k) \\ &\qquad + 2\,\mathrm{E}\Big[\,|V_1 + M_k|^\psi \mid V_1 > -M_k, V_2 \le -M_k\Big] \Pr(V_1 > -M_k, \%V_2 \le -M_k) \\ &\quad = \mathrm{E}\Big[|V_1 - V_2|^\psi \, \mathbf{1}_{\{V_1 > -M_k, V_2 > -M_k\}}\Big] + 2\,\mathrm{E}\Big[|V_1 + M_k|^\psi \, \mathbf{1}_{\{V_1 > -M_k, V_2 \le -M_k\}}\Big] \\ &\quad = \mathrm{E}\Big[\mathrm{E}\Big[|V_1 - V_2|^\psi \, \mathbf{1}_{\{V_1 > -M_k, V_2 > -M_k\}} \big| M_k\Big]\Big] \\ &\qquad + 2\,\mathrm{E}\Big[\mathrm{E}\Big[|V_1 + M_k|^\psi \, \mathbf{1}_{\{V_1 > -M_k, V_2 \le -M_k\}} \big| M_k\Big]\Big]. \end{aligned} \tag{9}$$

Remark 1 We assume $\mathrm{E}[X_i] < 0$ so that the G/G/1 queue is stable; and thus the distribution of W_k converges (regardless of W_0) to that of a finite r.v. W_∞ as $k \to \infty$ (see, e.g., [18, p. 103]). We also assume that the service times have a non-heavy-tailed distribution so that for some $c > 0$, we have $\mathrm{E}[e^{cS_i}] < \infty$; and so the m.g.f. of the X_i's, $\varphi(t) \equiv \mathrm{E}[e^{tX_i}]$, exists in the neighborhood $(-\infty, c)$ of the origin. Since $\varphi(t)$ is an m.g.f., it is a convex function. In addition, $\varphi'(0) = \mathrm{E}[X_i] < 0$; and it is straightforward to show that $\lim_{t \to +\infty} \varphi(t) = +\infty$. Let

$$\gamma \equiv \sup\{t > 0 : \varphi(t) < 1\},$$

which is positive and finite (except in the case $\Pr(W_\infty = 0) = 1$). Thus we see that γ is the unique positive quantity such that $\varphi(\gamma) = 1$ and $\varphi'(\gamma) > 0$. In the M/M/1 case, by the way, we obtain the closed-form result $\gamma = \lambda(1-\rho)/\rho$ (recall Example 1). Finally, in what follows, we assume the derivatives of $\varphi(t)$ at points 0 and γ are finite, i.e.,

$$\varphi'(0) = \mathrm{E}[X_i] > -\infty \qquad \text{and} \qquad \varphi'(\gamma) = \mathrm{E}[X_i e^{\gamma X_i}] < +\infty\,. \tag{10}$$

Note that γ can be used to formulate an exponential upper bound for the upper tail probability of W_∞:

$$\Pr(W_\infty \geq x) \leq e^{-\gamma x}\,, \quad \text{for all } x > 0\,; \tag{11}$$

see [18, Eq. (17) and the theorem on p. 106]. □

We state and prove two inequalities that will be useful in Theorem 1 below.

Lemma 1 *For two real numbers a and b, it is true that*

$$|a+b|^\psi \leq |a|^\psi + |b|^\psi \text{ if } 0 < \psi \leq 1, \tag{12}$$

and

$$|a+b|^\psi \leq 2^{\psi-1}(|a|^\psi + |b|^\psi) \text{ if } \psi > 1. \tag{13}$$

Proof From the observation that $|a+b|^\psi \leq ||a| + |b||^\psi$ for all $a, b \in \mathbb{R}$ and $\psi > 0$, it is straightforward to verify (12) when $ab = 0$; and otherwise, it is sufficient to verify (12) when $a = 1$ and $b \in (0,1)$. For $\psi \in [0,1]$, define the functions $\mathfrak{g}(\psi) \equiv (1+b)^\psi$ and $\mathfrak{h}(\psi) \equiv 1 + b^\psi$. We have $\mathfrak{g}(0) = 1$ and $\mathfrak{h}(0) = 2$, $\mathfrak{g}'(\psi) = \mathfrak{g}(\psi)\log(1+b) > 0$, $\mathfrak{h}'(\psi) = b^\psi \log b < 0$, and $\mathfrak{g}(1) = \mathfrak{h}(1) = 1 + b$. To verify Eq. (12), we prove by contradiction that $\mathfrak{g}(\psi) \leq \mathfrak{h}(\psi)$ for $\psi \in [0,1]$. Suppose that $\mathfrak{g}(\psi_0) > \mathfrak{h}(\psi_0)$ for some $\psi_0 \in (0,1)$. Since $\mathfrak{g}(\cdot)$ is increasing and $\mathfrak{h}(\cdot)$ is decreasing on $[0,1]$, we have $1 = \mathfrak{g}(1) > \mathfrak{g}(\psi_0) > \mathfrak{h}(\psi_0) > \mathfrak{h}(1) = 1$, a contradiction. Thus Eq. (12) holds. Finally to verify Eq. (13), we note that it is sufficient to verify (13) when $a, b > 0$; and in that case we let $\mathfrak{n} \equiv 2$, $\mathfrak{a}_1 \equiv a$ and $\mathfrak{a}_2 \equiv b$, $\mathfrak{r} \equiv 1$, and $\mathfrak{s} = \psi$ so that $\mathfrak{r} < \mathfrak{s}$. Then we can rewrite (13) in the form

$$\left(\frac{1}{\mathfrak{n}}\sum_{\nu=1}^{\mathfrak{n}} \mathfrak{a}_\nu^{\mathfrak{r}}\right)^{1/\mathfrak{r}} \leq \left(\frac{1}{\mathfrak{n}}\sum_{\nu=1}^{\mathfrak{n}} \mathfrak{a}_\nu^{\mathfrak{s}}\right)^{1/\mathfrak{s}},$$

which coincides with Eq. (2.9.1) of [16]; and thus Eq. (13) also holds. □

We are finally in a position to give our main result.

Theorem 1 *Consider the setup heretofore discussed. Suppose that* $E[X_i] < 0$ *for all* i *and that Eqs. (11) and (10) hold. Then the GMC condition (5) holds with a decay rate* $r = \exp\left[\int_0^1 \log \varphi(\gamma z)\, \mathrm{d}z\right] \in (0, 1)$.

Proof We start by replacing $-M_k$ in Eq. (9) with a deterministic nonnegative constant $x \geq 0$ and finding upper bounds for the inner conditional expectations. Along the way, we let $\mathcal{E} \sim \mathrm{Exp}(\gamma)$ be an exponential r.v. so that there is a stochastic order $W_\infty \leq_{\mathrm{st}} \mathcal{E}$ due to the bound in (11). Moreover, let $u(y) = |y - x|^\psi \mathbf{1}_{\{y>x\}}$, which is a nondecreasing function of y for all $x \geq 0$. Then

$$\begin{aligned}
\mathrm{E}\left[|W_\infty - x|^\psi \mathbf{1}_{\{W_\infty > x\}}\right] &= \mathrm{E}\left[u(W_\infty)\right] \\
&\leq \mathrm{E}\left[u(\mathcal{E})\right] \quad (W_\infty \leq_{\mathrm{st}} \mathcal{E} \text{ and } u(\cdot) \text{ is nondecreasing}) \\
&= \mathrm{E}\left[|\mathcal{E} - x|^\psi \mathbf{1}_{\{\mathcal{E}>x\}}\right] \\
&= \mathrm{E}\left[|\mathcal{E} - x|^\psi \,\middle|\, \mathcal{E} > x\right] \Pr(\mathcal{E} > x) \\
&= c(\psi)\, e^{-\gamma x}\,,
\end{aligned}$$

where $c(\psi) \equiv \mathrm{E}\left[|\mathcal{E} - x|^\psi \,\middle|\, \mathcal{E} > x\right] = \mathrm{E}[\mathcal{E}^\psi] = \Gamma(\psi + 1)/\gamma^\psi$ is a constant independent of x due to the memoryless property of the exponential distribution. Therefore, since V_1 and W_∞ have the same distribution, we have

$$\begin{aligned}
\mathrm{E}\left[|V_1 - x|^\psi \mathbf{1}_{\{V_1>x, V_2 \leq x\}}\right] &\leq \mathrm{E}\left[|V_1 - x|^\psi \mathbf{1}_{\{V_1>x\}}\right] \\
&= \mathrm{E}\left[|W_\infty - x|^\psi \mathbf{1}_{\{W_\infty>x\}}\right] \leq c(\psi)\, e^{-\gamma x}\,. \qquad (14)
\end{aligned}$$

Now, let $\kappa(\psi) \equiv \max\{1, 2^{\psi-1}\}$ for $\psi > 0$. Then

$$\begin{aligned}
&\mathrm{E}\left[|V_1 - V_2|^\psi \mathbf{1}_{\{V_1>x, V_2>x\}}\right] = \mathrm{E}\left[|(V_1 - x) - (V_2 - x)|^\psi \mathbf{1}_{\{V_1>x, V_2>x\}}\right] \\
&\quad \leq \mathrm{E}\left[\kappa(\psi)(|V_1 - x|^\psi + |V_2 - x|^\psi) \mathbf{1}_{\{V_1>x, V_2>x\}}\right] \quad \text{(by (12) and (13))} \\
&\quad \leq \kappa(\psi) \left\{\mathrm{E}\left[|V_1 - x|^\psi \mathbf{1}_{\{V_1>x, V_2>x\}}\right] + \mathrm{E}\left[|V_2 - x|^\psi \mathbf{1}_{\{V_1>x, V_2>x\}}\right]\right\} \\
&\quad = 2\kappa(\psi)\, \mathrm{E}\left[|V_1 - x|^\psi \mathbf{1}_{\{V_1>x, V_2>x\}}\right] \\
&\quad \leq 2\kappa(\psi)\, \mathrm{E}\left[|V_1 - x|^\psi \mathbf{1}_{\{V_1>x\}}\right] \\
&\quad \leq 2\kappa(\psi) c(\psi)\, e^{-\gamma x}\,,
\end{aligned}$$

where the last step follows from Eq. (14).

Since V_1 and V_2 are independent of M_k, we obtain from Eq. (9) that

$$\mathrm{E}\left[|W_k(V_1) - W_k^*(V_2)|^\psi\right] \leq 2c(\psi)(1 + \kappa(\psi))\, \mathrm{E}[e^{\gamma M_k}]\,. \qquad (15)$$

Next we find an upper bound for $\mathrm{E}[e^{\gamma M_k}]$ that decreases geometrically fast as $k \to \infty$. First note that

$$M_k \le \tilde{U}_{k+1} \equiv \frac{1}{k+1}\sum_{j=-1}^{k-1} U_j = \begin{cases} U_{-1} = X_{-1} = 0, & \text{if } k = 0, \\ \dfrac{1}{k+1}\displaystyle\sum_{j=-1}^{k-1}\sum_{i=-1}^{j} X_i = \dfrac{1}{k+1}\sum_{i=0}^{k-1}(k-i)X_i, & \text{if } k \ge 1. \end{cases}$$

This implies that $\mathrm{E}[e^{\gamma M_k}] \le \mathrm{E}[e^{\gamma \tilde{U}_{k+1}}]$ for $k \ge 0$. In particular, for $k = 0$, we have $\mathrm{E}[e^{\gamma M_0}] \le 1$; and for $k \ge 1$, we have

$$\begin{aligned} \mathrm{E}[e^{\gamma M_k}] &\le \mathrm{E}\Big[\exp\Big(\frac{\gamma}{k+1}\sum_{i=0}^{k-1}(k-i)X_i\Big)\Big] \\ &= \mathrm{E}\Big[\exp\Big(\frac{\gamma}{k+1}\sum_{i=1}^{k} i X_i\Big)\Big] \quad (X_i\text{'s are i.i.d.}) \\ &= \prod_{i=1}^{k} \mathrm{E}\Big[\exp\Big(\frac{i\gamma}{k+1}X_i\Big)\Big] \quad (X_i\text{'s are i.i.d.}) \\ &= \prod_{i=1}^{k} \varphi\Big(\frac{i\gamma}{k+1}\Big). \end{aligned} \tag{16}$$

Let $f(z) = \varphi(\gamma z)$ for $0 \le z \le 1$. Note that $f(0) = \varphi(0) = 1$ and $f(1) = \varphi(\gamma) = 1$. Since the m.g.f. φ is log-convex, f is log-convex too. On the other hand, due to Eq. (10), the logarithmic derivatives of f at 0 and 1 are $(\log f)'(0) = f'(0)/f(0) = \gamma\varphi'(0) = \gamma\mathrm{E}[X_i] \in (-\infty, 0)$ and $(\log f)'(1) = f'(1) = \gamma\varphi'(1) = \gamma\mathrm{E}[X_i e^{\gamma X_i}] \in (0, +\infty)$, respectively. Thus, $\log f(z) < 0$ for $0 < z < 1$, so that $\int_0^1 \log f(z)\,\mathrm{d}z < 0$; and then the quantity $r \equiv \exp\big[\int_0^1 \log f(z)\,\mathrm{d}z\big] \in (0, 1)$. Moreover, since f is a log-convex function, $(\log f)'$ is monotonically increasing; and so $\max_{z\in[0,1]}(\log f)'(z) = f'(1) < \infty$.

By Eq. (16) and the definition of $f(z)$, we have

$$\begin{aligned} \mathrm{E}[e^{\gamma M_k}] &\le \prod_{i=1}^{k} f\Big(\frac{i}{k+1}\Big) \\ &= \exp\left\{\sum_{i=1}^{k} \log f\Big(\frac{i}{k+1}\Big)\right\} \\ &= \left[\exp\left\{\frac{1}{k}\sum_{i=1}^{k} \log f\Big(\frac{i}{k+1}\Big)\right\}\right]^k \\ &\le \left[\exp\left\{\frac{1}{k+1}\sum_{i=1}^{k+1} \log f\Big(\frac{i}{k+1}\Big)\right\}\right]^k, \end{aligned} \tag{17}$$

where the final inequality holds because $\log f(1) = 0$ and $\log f(z) < 0$ for $z \in (0, 1)$.

By the error bound on Riemann sums for integrals, we have (see Eqs. (2.1.7)–(2.1.9) on p. 53 in [11])

$$\begin{aligned}\frac{1}{k+1}\sum_{i=1}^{k+1}\log f\left(\frac{i}{k+1}\right) - \int_0^1 \log f(z)\,\mathrm{d}z &\le \frac{1}{2(k+1)^2}\sum_{i=1}^{k+1}\sup_{z\in[\frac{i-1}{k+1},\frac{i}{k+1}]}(\log f)'(z)\\ &\le \frac{1}{2(k+1)}\max_{i=1,\ldots,k+1}\sup_{z\in[\frac{i-1}{k+1},\frac{i}{k+1}]}(\log f)'(z)\\ &= \frac{\max_{z\in[0,1]}(\log f)'(z)}{2(k+1)}\\ &= \frac{f'(1)}{2(k+1)} \quad \text{for } k = 0, 1, \ldots . \end{aligned} \tag{18}$$

Therefore, Eqs. (17)–(18) and some algebra give us the following upper bound on $\mathrm{E}[e^{\gamma M_k}]$:

$$\begin{aligned}\mathrm{E}[e^{\gamma M_k}] &\le \left[\exp\left\{\frac{1}{k+1}\sum_{i=1}^{k+1}\log f\left(\frac{i}{k+1}\right)\right\}\right]^k\\ &\le \left[\exp\left\{\int_0^1 \log f(z)\,\mathrm{d}z + \frac{f'(1)}{2(k+1)}\right\}\right]^k\\ &= r^k \exp\left\{\frac{k f'(1)}{2(k+1)}\right\}\\ &\le \begin{cases} 1, & \text{for } k = 0,\\ r^k e^{f'(1)/2}, & \text{for } k = 1, 2, \ldots . \end{cases}\end{aligned} \tag{19}$$

It follows from Eqs. (15) and (19) that there is a sufficiently large positive quantity C depending on $f'(1)$, $c(\psi)$, and $\kappa(\psi)$ for which

$$\mathrm{E}\left[\,\left|W_k(V_1) - W_k^*(V_2)\right|^{\psi}\,\right] \le C r^k \quad \text{for } k = 0, 1, \ldots . \quad \square \tag{20}$$

□

Example 2: For the special case of the M/M/1 queue, it is possible to obtain a closed-form formula for the decay rate r in (20) (as a function of ρ). It is also possible to explicitly show that the two conditions, $r = \exp\left[\int_0^1 \log f(z)\,\mathrm{d}z\right] \in (0, 1)$ and $f'(1) < \infty$, are satisfied for any $\rho \in (0, 1)$. To do so, note that

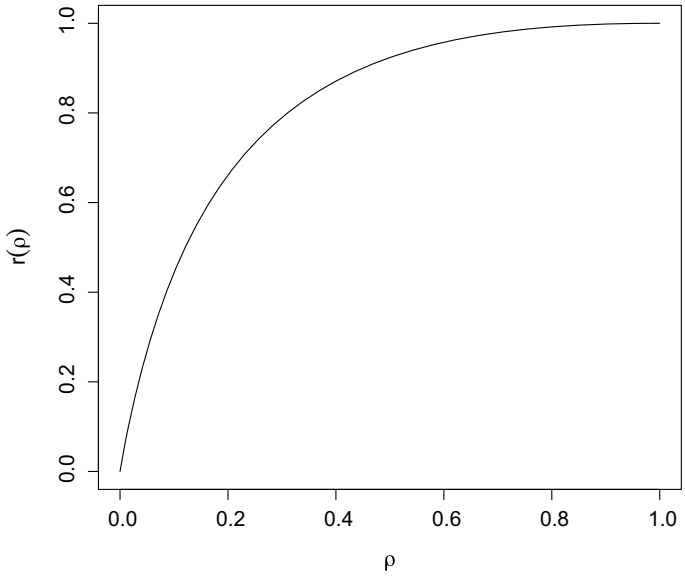


Fig. 1 Plot of $r(\rho)$ for an M/M/1 queue

$$f(z) = \mathrm{E}[\exp(z\gamma X_i)] \;=\; \mathrm{E}[\exp(z\gamma S_i)]\,\mathrm{E}[\exp(-z\gamma T_i)] \;\text{ for }\; z \in [0,1] \;\text{ and }\; i \geq 0$$
$$= \left[\frac{1}{1 - z(1-\rho)}\right]\left[\frac{1}{1 + z(1-\rho)/\rho}\right] > 0 \;\text{ for }\; z \in [0,1],$$

since $\rho \in (0,1)$ and $z \in [0,1]$ so that $0 \leq z(1-\rho) < 1$. The function f is symmetric on $[0,1]$, i.e., $f(z) = f(1-z)$ for $z \in [0,1]$; and $f(0) = f(1) = 1$. In addition, it can be shown that

$$r(\rho) = \exp\left[\int_0^1 \log f(z)\,\mathrm{d}z\right] = e^2 \rho^{\frac{1+\rho}{1-\rho}} \quad \text{and} \quad f'(1) = \frac{(1-\rho)^2}{\rho}. \tag{21}$$

Further, we see that (i) for each $\rho \in (0,1)$, we have $0 < r(\rho) < 1$ (cf. Fig. 1); (ii) $\frac{\mathrm{d}}{\mathrm{d}\rho} r(\rho) > 0$ for $\rho \in (0,1)$; (iii) $r(\rho) \to 0$ as $\rho \to 0$; and (iv) $r(\rho) \to 1$ as $\rho \to 1$. Also observe that for each $\rho \in (0,1)$, we have $f'(1) < \infty$. □

3 Monte Carlo Results

The purpose of this section is to illustrate empirical and robustness properties of $\mathrm{E}[|W_k(V_1) - W_k^*(V_2)|^\psi]$ on various simple, single-server queueing systems, where V_1 and V_2 are i.i.d. r.v.'s having the same distribution as the steady-state waiting time.

In particular, we check to see whether or not the decay of $\mathrm{E}[|W_k(V_1) - W_k^*(V_2)|^\psi]$ adheres to a geometric rate as k becomes large.

3.1 M/M/1 Queue

We plot Monte Carlo estimates of $\log \mathrm{E}[|W_k(V_1) - W_k^*(V_2)|^\psi]$ in Fig. 2 for $\psi = 1.5$ and $k = 1, \ldots, 800$, based on one million paths of waiting time pairs $\{W_k(V_1), W_k^*(V_2)\}, k = 1, \ldots, 800$, generated by Lindley's recursion, where we used the same paths for each k. The arrival and service rates are $\lambda = 0.8$ and $\mu = 1$, respectively. The red line is the fitted linear regression line, the R^2 value of which is 0.988, indicating a reasonable fit (and thus, the desired approximately geometric decay). Also, the estimated slope is -0.01542 so that the estimated rate is $e^{-0.01542} = 0.9847$. Therefore, since the empirical plot is roughly linear and the estimated rate is less than one, we can conclude that the decay rate seems to be geometric as suggested by Theorem 1. Furthermore, from Eq. (21), the asymptotic rate of the upper bound in Theorem 1 is $r = e^2(0.8)^9 = 0.9917$, which is higher but close to the estimated rate.

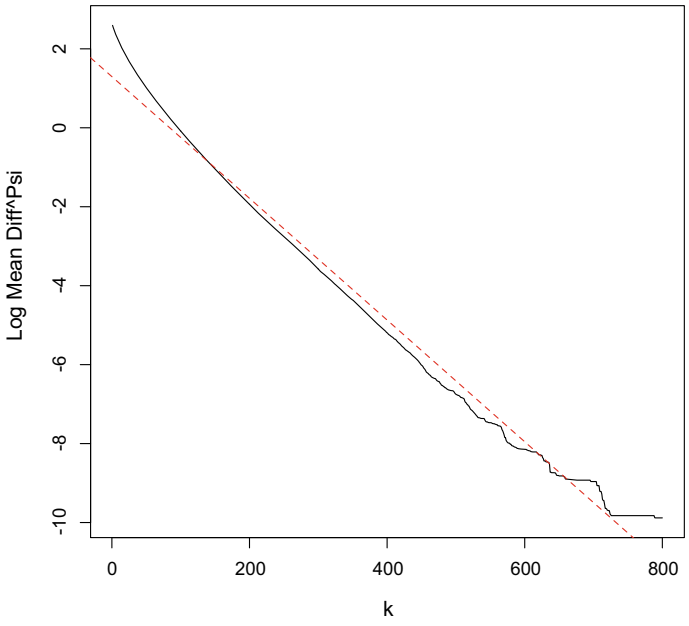


Fig. 2 Monte Carlo estimates of $\log \mathrm{E}\big[|W_k(V_1) - W_k^*(V_2)|^{1.5}\big]$ for an M/M/1 queue with $\lambda = 0.8$ and $\mu = 1$

3.2 M/G/1 Queues

To simulate $|W_k(V_1) - W_k^*(V_2)|^\psi$, $k = 1, 2, \ldots$, we first need to sample V_1 and V_2 from the stationary waiting time distribution. For M/G/1 queues, it is possible to sample from the "exact" stationary distribution. Let $F_S(x)$ denote the c.d.f. of the random service time S. According to the Pollaczek–Khinchine formula, the stationary waiting time is equal in distribution to $\sum_{i=1}^{N} Y_i$, where the Y_i's are i.i.d. r.v.'s having the probability density function (p.d.f.)

$$f_Y(x) = \frac{1 - F_S(x)}{\mathrm{E}[S]},$$

and N is a geometric r.v., which is independent of the Y_i's, with a success probability $1 - \rho$ and the probability mass function (p.m.f.) $\Pr(N = u) = \rho^u(1 - \rho)$, $u = 0, 1, 2, \ldots$; see e.g., p. 21 of [7]. So, in order to sample from the exact stationary distribution, what we need to do is to generate a geometric r.v. N and then N i.i.d. copies of Y. We can subsequently sample from the density $f_Y(x)$ by using standard methods (as described in what follows).

3.2.1 M/H$_2$/1 Queue

In an M/H$_2$/1 queue, the service time S follows a hyperexponential distribution with p.d.f.

$$f_S(x) = p\mu_1 e^{-\mu_1 x} + (1 - p)\mu_2 e^{-\mu_2 x}, \quad \text{for } x \geq 0,$$

where $0 < p < 1$ and $\mu_1, \mu_2 > 0$. So,

$$1 - F_S(x) = pe^{-\mu_1 x} + (1 - p)e^{-\mu_2 x}$$

and

$$\mathrm{E}[S] = \frac{p}{\mu_1} + \frac{1 - p}{\mu_2}.$$

The density of Y is

$$f_Y(x) = q\mu_1 e^{-\mu_1 x} + (1 - q)\mu_2 e^{-\mu_2 x},$$

where

$$q = \frac{p/\mu_1}{p/\mu_1 + (1 - p)/\mu_2}.$$

Thus, Y is also a hyperexponential r.v., and one can easily generate an i.i.d. sample of Y's. We can then simulate the stationary waiting time by computing $\sum_{i=1}^{N} Y_i$, where N is a shifted geometric, as described in Sect. 3.2. We can alternatively do

so via a more-direct route: Let $\Gamma(k, \lambda)$ denote the gamma distribution with shape parameter k and rate parameter λ; let $\text{Bin}(n, p)$ denote the binomial distribution with number of trials n and success probability p; and let $\text{Geom}(p)$ denote the "shifted" geometric distribution with success probability p and p.m.f. $f(i) = (1-p)^i p$, for $i = 0, 1, 2, \ldots$. Also, let $N_1 \sim \text{Bin}(N, q)$ and $N_2 = N - N_1$, where $N \sim \text{Geom}(1-\rho)$. If $N = 0$, then $\sum_{i=1}^{N} Y_i = 0$; and otherwise,

$$\sum_{i=1}^{N} Y_i \sim X_1 + X_2 \,,$$

where $X_1 \sim \Gamma(N_1, \mu_1)$ and $X_2 \sim \Gamma(N_2, \mu_2)$ are two independent gamma r.v.'s. So, the steps for simulating from the stationary waiting time distribution are

1. Generate a geometric r.v., $N \sim \text{Geom}(1-\rho)$. If $N = 0$, return 0, else continue.
2. Generate a binomial r.v., $N_1 \sim \text{Bin}(N, q)$ and set $N_2 = N - N_1$.
3. Generate $X_1 \sim \Gamma(N_1, \mu_1)$ and $X_2 \sim \Gamma(N_2, \mu_2)$. Return stationary waiting time $X_1 + X_2$.

Note that [17] showed that the stationary waiting time distribution of a G/G/1 queue with hyperexponential service and interarrival times also has a hyperexponential structure.

Moreover, for an M/H_2/1 queue, it is possible to calculate the decay rate $r = \exp\left[\int_0^1 \log f(z)\,\mathrm{d}z\right]$ of the upper bound in (20), at least numerically. The m.g.f. of $X_i = S_i - T_i$ is

$$\varphi(t) = \frac{\lambda}{\lambda + t}\left[p\left(\frac{\mu_1}{\mu_1 - t}\right) + (1-p)\left(\frac{\mu_2}{\mu_2 - t}\right)\right] \quad \text{for } t < \min\{\mu_1, \mu_2\}\,.$$

The positive solution of $\varphi(\gamma) = 1$ is

$$\gamma = \frac{1}{2}\left(-\lambda + \mu_1 + \mu_2 - \sqrt{(\lambda + \mu_1 - \mu_2)^2 + 4\lambda(-\mu_1 + \mu_2)p}\right).$$

The integral $\int_0^1 \log \varphi(\gamma z)\,\mathrm{d}z$ has no closed-form solution, but can be calculated numerically.

Figure 3 plots Monte Carlo estimates of $\log \mathrm{E}[|W_k(V_1) - W_k^*(V_2)|^\psi]$ for $\psi = 1.5$ and $k = 1, \ldots, 1500$, based on 100,000 replications for each k. The parameters of the hyperexponential distribution are $p = (5 + \sqrt{15})/10 \approx 0.8873$, $\mu_1 = 2.5\,p$, and $\mu_2 = 2.5(1-p)$, and the arrival rate is $\lambda = 1$. With these parameter values, $\mathrm{E}[S] = 0.8$, and so $\rho = \lambda \mathrm{E}[S] = 0.8$. In addition, the stationary mean waiting time is 8. The red line again depicts the fitted regression line, with associated $R^2 = 0.974$. Also, the estimated slope is -0.005986, so that the estimated decay rate is $e^{-0.005986} = 0.994$. For the selected parameter values, we have

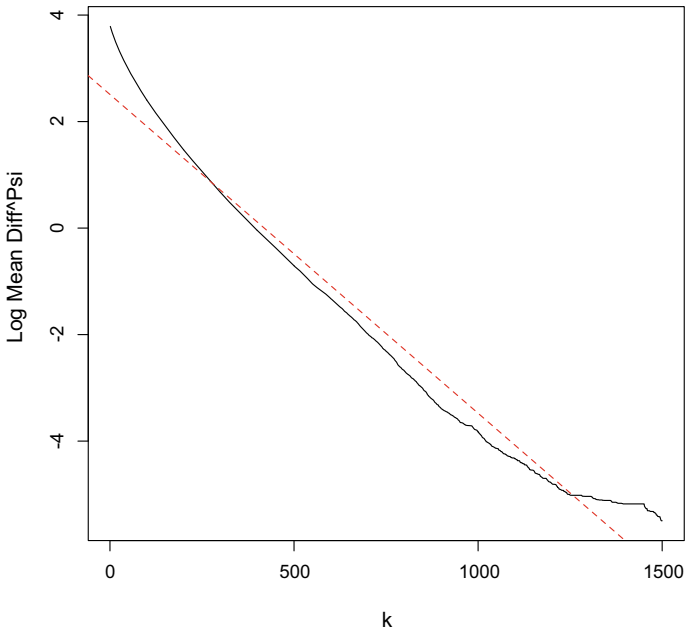


Fig. 3 Monte Carlo estimates of $\log \mathrm{E}\left[|W_k(V_1) - W_k^*(V_2)|^{1.5}\right]$ for an M/H$_2$/1 queue with $\lambda = 1$ and $\rho = 0.8$

$$f(z) = \varphi(\gamma z) = \frac{8\left[5 + 4\left(-3 + \sqrt{7}\right)z\right]}{40 + 30\left(-3 + \sqrt{7}\right)z + 12\left(-8 + 3\sqrt{7}\right)z^2 + \left(90 - 34\sqrt{7}\right)z^3}\,.$$

By numerical integration, the decay rate of the upper bound is obtained as $r = \exp\left[\int_0^1 \log f(z)\,\mathrm{d}z\right] = 0.997$. Thus, as in the M/M/1 case, we see an empirical confirmation of Theorem 1. This is not surprising since all the conditions of Theorem 1 are satisfied for the M/H$_2$/1 queue as well.

3.2.2 M/G/1 with Lognormal Service Time

Suppose that the service time S follows a lognormal distribution with parameters μ and σ, i.e., $\log S \sim \mathrm{Nor}(\mu, \sigma^2)$. Then

$$1 - F_S(x) = \Phi\left[-(\log x - \mu)/\sigma\right]\,,$$

where $\Phi(\cdot)$ is the c.d.f. of the standard normal distribution, and

$$\mathrm{E}[S] = e^{\mu + \sigma^2/2}\,.$$

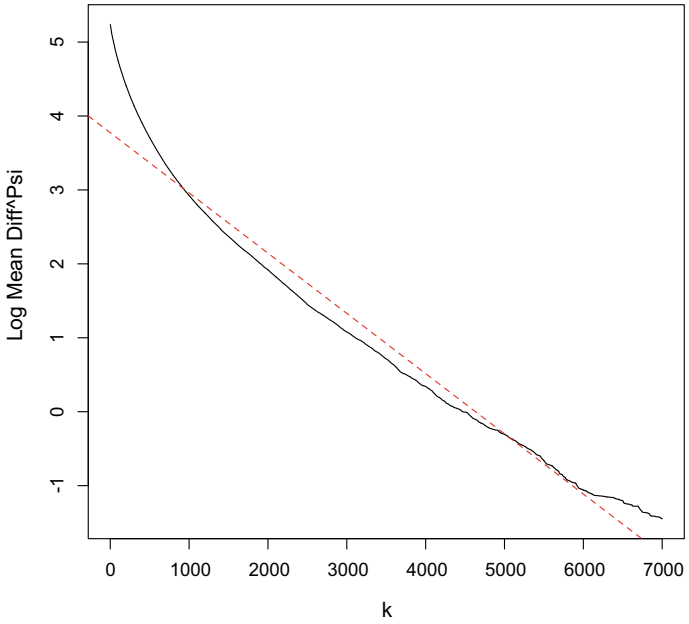


Fig. 4 Monte Carlo estimates of $\log \mathrm{E}\left[|W_k(V_1) - W_k^*(V_2)|^{1.5}\right]$ for an M/G/1 queue with a lognormal service time and utilization $\rho = 0.8$

The density of Y is therefore

$$f_Y(x) = e^{-\mu-\sigma^2/2}\, \Phi\left[-(\log x - \mu)/\sigma\right] .$$

To generate samples from $f_Y(x)$, the numerical inversion method of [12] can be used.

Figure 4 plots the Monte Carlo estimates of $\log \mathrm{E}[|W_k(V_1) - W_k^*(V_2)|^{\psi}]$ for $\psi = 1.5$ and $k = 1, \ldots, 7{,}000$, based on 100,000 replications for each k. The parameter values of the lognormal distribution are $\mu = -1.374436$ and $\sigma = 1.517427$ and the arrival rate is $\lambda = 1$. With these parameter values, $\mathrm{E}[S] = 0.8$, and so $\rho = \lambda \mathrm{E}[S] = 0.8$. In addition, the expected value of the stationary waiting time is 16. The red fitted regression line corresponds to an R^2 value of 0.969. The estimated slope is -0.0008147 and the estimated decay rate is $e^{-0.0008147} = 0.9992$, which is very close to one. Since the lognormal distribution is a heavy-tailed distribution, whose m.g.f. is undefined for positive arguments, the assumptions of Theorem 1 are violated. Therefore, we have no theoretical result stating that the GMC condition is satisfied for this queueing system. Also, note that the R^2 value and estimated decay rate are worse than those of M/M/1 and M/H_2/1 examples. So, we have weaker empirical evidence regarding the adherence to the GMC condition.

3.2.3 M/G/1 with Pareto Service Time

Finally, suppose that S follows a Pareto distribution with shape parameter $\alpha > 1$, the complementary c.d.f.

$$1 - F_S(x) = 1/(1+x)^{\alpha}\,,$$

and the expected value

$$\mathrm{E}[S] = 1/(\alpha - 1)\,.$$

So, the density of Y is

$$f_Y(x) = (\alpha - 1)(1+x)^{-\alpha}\,, \quad \text{for } x \geq 0\,.$$

Its c.d.f. is

$$F_Y(x) = 1 - (1+x)^{1-\alpha}\,, \quad \text{for } x \geq 0\,,$$

and the inverse c.d.f. is

$$F_Y^{-1}(u) = -1 + (1-u)^{1/(1-\alpha)}\,, \quad \text{for } 0 < u < 1\,.$$

We can generate an i.i.d. sample of Y by using the above inverse c.d.f. and an i.i.d. sample of uniforms.

Figure 5 depicts the Monte Carlo estimates of $\log \mathrm{E}[|W_k(V_1) - W_k^*(V_2)|^{\psi}]$ for $\psi = 1.5$ and $k = 1, \ldots, 11{,}000$, based on 100,000 replications for each k. The shape parameter of the Pareto distribution is $\alpha = 2.25$ and the arrival rate is $\lambda = 1$. With these parameter values, $\mathrm{E}[S] = 0.8$ and so $\rho = \lambda \mathrm{E}[S] = 0.8$. In addition, the expected value of the stationary waiting time is 16, and its variance is infinite. The red fitted regression line corresponds to an $R^2 = 0.943$. The estimated slope is $-7.665\text{E-}05$, the estimated rate is $e^{-7.665\mathrm{E}-05} = 0.9999234$, which is extremely close to unity, and the linear fit appears to be exceptionally poor. In addition, the Pareto distribution is heavy-tailed, so that the assumptions of Theorem 1 are violated. In any case, it is not surprising to see that empirical evidence for this example appears to mitigate against a geometric decay rate, at least compared to the M/M/1 and M/H_2/1 examples.

4 Conclusions

We have proved that the GMC condition holds for the queue-waiting-time process in a G/G/1 queueing system whose service-time distribution is non-heavy-tailed. We have also demonstrated the use of empirical techniques for checking that the GMC condition approximately holds for the G/G/1 queue-waiting-time process based on more-general service-time distributions. Such results are useful since the GMC condition enables certain output-analysis procedures to work reliably when they are applied to a broad class of output processes that are generated by a simulation in

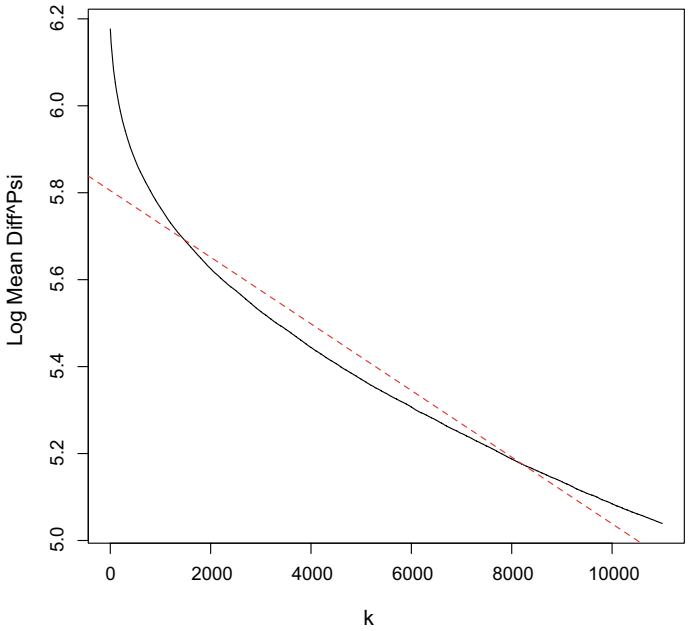


Fig. 5 Monte Carlo estimates of $\log \mathrm{E}\left[|W_k(V_1) - W_k^*(V_2)|^{1.5}\right]$ for an M/G/1 queue with a Pareto service time and utilization $\rho = 0.8$

steady-state operation. In particular, the GMC assumption underlies recent procedures that are based on the methods of batch means and standardized time series and that are designed to deliver asymptotically valid point and CI estimators of the steady-state process mean or selected quantiles. Whereas the GMC condition can be empirically checked in practice, the usual moment and mixing conditions are much more problematic to check as detailed in [4, Remark 2]. Among promising directions for future work, we are currently investigating the theoretical connections between the GMC and FCLT conditions; and we are also investigating the possible applicability of the GMC condition to sojourn (cycle) times of customers in certain types of queueing networks.

Acknowledgements We thank Pierre L'Ecuyer not only for his remarkable contributions to the theory and practice of computer simulation over the past four decades, but also for his equally remarkable contributions as an editor and reviewer in a broad diversity of scientific disciplines, where his work has been applied extensively.

References

1. Aktaran-Kalaycı, T., Alexopoulos, C., Argon, N.T., Goldsman, D., Wilson, J.R.: Exact expected values of variance estimators in steady-state simulation. Nav. Res. Logist. **54**(4), 397–410 (2007)
2. Alexopoulos, C., Argon, N.T., Goldsman, D., Tokol, G., Wilson, J.R.: Overlapping variance estimators for simulation. Oper. Res. **55**(6), 1090–1103 (2007)
3. Alexopoulos, C., Boone, J.H., Goldsman, D., Lolos, A., Dingeç, K.D., Wilson, J.R.: Steady-state quantile estimation using standardized time series. In: K.H. Bae, B. Feng, S. Kim, L. Lazarova-Molnar, Z. Zheng, T. Roeder, R. Thiesing (eds.) Proceedings of the 2020 Winter Simulation Conference, pp. 289–300. Institute of Electrical and Electronics Engineers, Piscataway, New Jersey (2020)
4. Alexopoulos, C., Dingeç, K.D., Goldsman, D., Lolos, A., Mokashi, A.C., Wilson, J.R.: Steady-state quantile estimation using standardized time series. Technical report (2022). https://people.engr.ncsu.edu/jwilson/files/stsms-112821.pdf. Accessed 10th Feb 2022
5. Alexopoulos, C., Goldsman, D., Mokashi, A.C., Tien, K.W., Wilson, J.R.: Sequest: a sequential procedure for estimating quantiles in steady-state simulations. Oper. Res. **67**(4), 1162–1183 (2019). https://people.engr.ncsu.edu/jwilson/files/sequest19or.pdf. Accessed 7th Sept 2019
6. Alexopoulos, C., Goldsman, D., Wilson, J.R.: A new perspective on batched quantile estimation. In: C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, A.M. Uhrmacher (eds.) Proceedings of the 2012 Winter Simulation Conference, pp. 190–200. Institute of Electrical and Electronics Engineers, Piscataway, New Jersey (2012)
7. Asmussen, S., Glynn, P.W.: Stochastic Simulation: Algorithms and Analysis. Springer, New York (2007)
8. Beran, J.: Statistics for Long-Memory Processes. Chapman & Hall/CRC, Boca Raton, Florida (1994)
9. Bollerslev, T.: Generalized autoregressive conditional heteroskedasticity. J. Econ. **31**, 307–327 (1986)
10. Damerdji, H.: Strong consistency of the variance estimator in steady-state simulation output analysis. Math. Oper. Res. **19**, 494–512 (1994)
11. Davis, P.J., Rabinowitz, P.: Methods of Numerical Integration, 2nd edn. Academic Press, Orlando, FL (1984)
12. Derflinger, G., Hörmann, W., Leydold, J.: Random variate generation by numerical inversion when only the density is known. ACM Trans. Model. Comput. Simul. **20**(18), 1–25 (2010)
13. Dingeç, K., Goldsman, D., Alexopoulos, C., Lolos, A., Wilson, J.R.: Geometric-moment contraction, stationary processes, and their indicator processes. Technical report, Gebze Technical University, Georgia Institute of Technology, North Carolina State University (2022). https://people.engr.ncsu.edu/jwilson/files/gmcind-030822.pdf. Accessed 8th Feb 2022
14. Engle, R.F.: Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. Econometrica **50**(4), 987–1007 (1982)
15. Glynn, P.W., Whitt, W.: Estimating the asymptotic variance with batch means. Oper. Res. Lett. **10**, 431–435 (1991)
16. Hardy, G.H., Littlewood, J.E., Polya, G.: Inequalities, 2nd edn. Cambridge University Press, Cambridge (1952)
17. Keilson, J., Machihara, F.: Hyperexponential waiting time structure in hyperexponential normal upper H Subscript normal upper K Baseline divided by normal upper H Subscript normal upper L Baseline divided by $1H_K/H_L/1$ systems. J. Oper. Res. Soc. Jpn. **28**(3), 242–251 (1985)
18. Kingman, J.F.: Inequalities in the theory of queues. J. Roy. Stat. Soc. B **32**(1), 102–110 (1970)
19. Li, W.K., Ling, S., McAleer, M.: Recent theoretical results for time series models with GARCH errors. J. Econ. Surv. **16**(3), 245–269 (2002)
20. Mori, M.: Transient behaviour of the mean waiting time and its exact forms in M/M/1 and M/D/1. J. Oper. Res. Soc. Jpn. **19**(1), 14–31 (1976)
21. Nicholls, D.F., Quinn, B.G.: Random Coefficient Autoregressive Models: An Introduction. Springer, New York (1982)

22. Rolski, T., Schmidli, H., Schmidt, V., Teugels, J.: Stochastic Processes for Insurance and Finance. Wiley, New York (1999)
23. Shao, X., Wu, W.B.: Asymptotic spectral theory for nonlinear time series. Ann. Stat. **35**(4), 1773–1801 (2007)
24. Tong, H.: Nonlinear Time Series: A Dynamical System Approach. Oxford University Press, New York (1990)
25. Wu, W.B.: On the Bahadur representation of sample quantiles for dependent sequences. Ann. Stat. **33**(4), 1934–1963 (2005)
26. Wu, W.B., Shao, X.: Limit theorems for iterated random functions. J. Appl. Probab. **41**, 425–436 (2004)
27. Wu, W.B., Woodroofe, M.: A central limit theorem for iterated random functions. J. Appl. Probab. **37**(3), 748–755 (2000)

Tractability of Approximation in the Weighted Korobov Space in the Worst-Case Setting

Adrian Ebert, Peter Kritzer, and Friedrich Pillichshammer

Abstract In this paper we consider L_p-approximation, $p \in \{2, \infty\}$, of periodic functions from weighted Korobov spaces. In particular, we discuss tractability properties of such problems, which means that we aim to relate the dependence of the information complexity on the error demand ε and the dimension d to the decay rate of the weight sequence $(\gamma_j)_{j\geq 1}$ assigned to the Korobov space. Some results have been well known since the beginning of this millennium, others have been proven quite recently. We give a survey of these findings and will add some new results on the L_∞-approximation problem. To conclude, we give a concise overview of results and collect a number of interesting open problems.

Keywords Approximation · Worst-case error · Average-case error · Tractability · Weighted Korobov space

1 Introduction

In this paper we consider L_p-approximation, where $p \in \{2, \infty\}$, of periodic functions from a weighted Korobov space with smoothness parameter α from the viewpoint of Information-Based Complexity. In particular, we study the information complexity $n(\varepsilon, d)$ of these problems, which is the minimal number of information evaluations required to push the approximation error below a certain error demand $\varepsilon \in (0, 1)$ for problems in dimension $d \in \mathbb{N}$. The information classes considered are the class Λ^{all} consisting of arbitrary continuous linear functionals and the class Λ^{std} consisting of

A. Ebert · P. Kritzer
Johann Radon Institute for Computational and Applied Mathematics (RICAM), Austrian Academy of Sciences, Altenbergerstr. 69, 4040 Linz, Austria
e-mail: peter.kritzer@oeaw.ac.at

F. Pillichshammer (✉)
Institut für Finanzmathematik und Angewandte Zahlentheorie, Johannes Kepler Universität Linz, Altenbergerstr. 69, 4040 Linz, Austria
e-mail: friedrich.pillichshammer@jku.at

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_7

point evaluations only. Furthermore, we will distinguish between the absolute and the normalized error criterion in the worst-case setting.

If the information complexity $n(\varepsilon, d)$ grows exponentially in d for d tending to infinity, the problem is said to suffer from the curse of dimensionality. Otherwise, for sub-exponential growth rates, the problem is said to be tractable. Initially, only the notions of polynomial and strong polynomial tractability were introduced and studied in the literature. An extensive overview of tractability of multivariate problems can be found in the trilogy [10–12].

For weighted function classes, one assigns real numbers (weights) to the coordinates in order to model varying influence of the single variables on the approximation problem, and one is interested in (matching) necessary and sufficient conditions on the weights which guarantee tractability. In the particular case of L_2-approximation for the weighted Korobov space, matching conditions can be found in the paper [14] by Wasilkowski and Woźniakowski for the information class Λ^{all} and in the paper [9] by Novak, Sloan, and Woźniakowski for Λ^{std}. For L_∞-approximation, results on (strong) polynomial tractability are due to Kuo, Wasilkowski, and Woźniakowski; see [5] for Λ^{all} and [6] for Λ^{std}.

After (strong) polynomial tractability, more and finer notions of tractability have been introduced with the aim of obtaining a more detailed and clearer picture of the tractability of multivariate problems. Nowadays, there is a variety of finer notions of tractability comprising quasi-polynomial tractability, weak tractability, and uniform weak tractability. The exact definitions will be given in Definition 1. Based on this development, many multivariate problems need to be reconsidered in order to classify them further with respect to the newer notions of tractability. This has been done recently in [2] for the problem of L_2-approximation for weighted Korobov spaces. These results will be summarized in Sect. 3. In the present paper we shall also study the L_∞-case. We derive necessary and sufficient conditions for several notions of tractability (see Sect. 4). The presented conditions are tight, but unfortunately do not match exactly. Here, some problems remain open.

In Sect. 5 we give a concise survey of the current state of research in tractability theory of approximation in weighted Korobov spaces and formulate some interesting open questions.

Notation and basic definitions will be introduced in the following section.

2 Basic Definitions

2.1 Function Space Setting

The Korobov space $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$ with weight sequence $\boldsymbol{\gamma} = (\gamma_j)_{j\geq 1}$ in $\mathbb{R}^+$ is a reproducing kernel Hilbert space with kernel function $K_{d,\alpha,\boldsymbol{\gamma}} : [0,1]^d \times [0,1]^d \to \mathbb{R}$ given by

$$K_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{x}, \boldsymbol{y}) := \sum_{\boldsymbol{h}\in\mathbb{Z}^d} r_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{h}) \exp(2\pi \mathrm{i}\, \boldsymbol{h}\cdot(\boldsymbol{x}-\boldsymbol{y})),$$

where by "$\cdot$" we denote the usual dot product. The corresponding inner product and norm are given by

$$\langle f, g\rangle_{d,\alpha,\boldsymbol{\gamma}} := \sum_{\boldsymbol{h}\in\mathbb{Z}^d} \frac{1}{r_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{h})}\, \widehat{f}(\boldsymbol{h})\,\overline{\widehat{g}(\boldsymbol{h})} \quad \text{and} \quad \|f\|_{d,\alpha,\boldsymbol{\gamma}} = \sqrt{\langle f, f\rangle_{d,\alpha,\boldsymbol{\gamma}}}\,.$$

Here, the Fourier coefficients of a function $f \in \mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$ are given by

$$\widehat{f}(\boldsymbol{h}) = \int_{[0,1]^d} f(\boldsymbol{x}) \exp(-2\pi\,\mathrm{i}\,\boldsymbol{h}\cdot\boldsymbol{x})\mathrm{d}\boldsymbol{x},$$

and the decay function equals, for $\boldsymbol{h} = (h_1, \ldots, h_d)$, $r_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{h}) = \prod_{j=1}^{d} r_{\alpha,\gamma_j}(h_j)$, with $\alpha > 1$ (the so-called smoothness parameter of the space), and

$$r_{\alpha,\gamma}(h) := \begin{cases} 1 & \text{for } h = 0\,, \\ \gamma/|h|^\alpha & \text{for } h \in \mathbb{Z}\setminus\{0\}. \end{cases}$$

The kernel $K_{d,\alpha,\boldsymbol{\gamma}}$ is well-defined for $\alpha > 1$ and for all $\boldsymbol{x}, \boldsymbol{y} \in [0,1]^d$, since

$$|K_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{x},\boldsymbol{y})| \le \sum_{\boldsymbol{h}\in\mathbb{Z}^d} r_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{h}) = \prod_{j=1}^{d} \big(1 + 2\zeta(\alpha)\gamma_j\big) < \infty,$$

where ζ is the Riemann zeta function (note that $\zeta(\alpha) < \infty$ since $\alpha > 1$).

Furthermore, we assume here that the weights are ordered and satisfy

$$1 \ge \gamma_1 \ge \gamma_2 \ge \cdots > 0.$$

The weights $\boldsymbol{\gamma}$ and the smoothness parameter α are parameters of the Korobov space $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$.

The weighted Korobov space is a popular reference space for quasi-Monte Carlo rules, in particular for lattice rules. See, e.g., [7, Chap. 4] or [10, Appendix A] and the references therein.

2.2 Approximation in $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$

In this paper we consider L_p-approximation of functions from the weighted Korobov space $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$ for fixed weights $\boldsymbol{\gamma}$ and fixed smoothness parameter α for $p \in \{2, \infty\}$. We consider the operator $\mathrm{APP}_{d,p} : \mathcal{H}_{d,\alpha,\boldsymbol{\gamma}} \to L_p([0,1]^d)$ with $\mathrm{APP}_{d,p}(f) = f$ for all $f \in \mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$. Note that, strictly speaking, also this operator depends on α and on $\boldsymbol{\gamma}$, but as these parameters are considered to be fixed we do not include them explicitly in the notation and simply write $\mathrm{APP}_{d,p}$ rather than $\mathrm{APP}_{d,p,\alpha,\boldsymbol{\gamma}}$. The operator $\mathrm{APP}_{d,p}$ is the embedding from the weighted Korobov space $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$ to the space $L_p([0,1]^d)$.

In order to approximate $\mathrm{APP}_{d,p}$ with respect to the L_p-norm $\|\cdot\|_{L_p}$ over $[0,1]^d$, $p \in \{2,\infty\}$, it suffices to employ linear algorithms $A_{n,d}$ that use n information evaluations and are of the form

$$A_{n,d}(f) = \sum_{i=1}^{n} T_i(f)\, g_i \quad \text{for } f \in \mathcal{H}_{d,\alpha,\boldsymbol{\gamma}} \tag{1}$$

with functions $g_i \in L_p([0,1]^d)$ and bounded linear functionals $T_i \in \mathcal{H}^*_{d,\alpha,\boldsymbol{\gamma}}$ for $i = 1,\dots,n$; see [1] and also [8, 10]. We will assume that the functionals T_i belong to some permissible class of information Λ. In particular, we study the class Λ^{all} consisting of the entire dual space $\mathcal{H}^*_{d,\alpha,\boldsymbol{\gamma}}$ and the class Λ^{std}, which consists only of point evaluation functionals. Recall that $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$ is a reproducing kernel Hilbert space, which means that point evaluations are continuous linear functionals and therefore Λ^{std} is a subclass of Λ^{all}. With some abuse of notation we will write $A_{n,d} \in \Lambda$ if $A_{n,d}$ is a linear algorithm of the form (1) using information from the class Λ.

We remark that in both cases $p = 2$ and $p = \infty$, the embedding operator $\mathrm{APP}_{d,p}$ is continuous for all $d \in \mathbb{N}$, which can be seen as follows.

- For $p = 2$, we have for all $f \in \mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$ that

$$\begin{aligned}\|\mathrm{APP}_{d,2}(f)\|_{L_2}^2 = \|f\|_{L_2}^2 &= \sum_{\boldsymbol{h}\in\mathbb{Z}^d} |\widehat{f}(\boldsymbol{h})|^2 \\ &\le \sum_{\boldsymbol{h}\in\mathbb{Z}^d} \frac{1}{r_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{h})} |\widehat{f}(\boldsymbol{h})|^2 = \|f\|_{d,\alpha,\boldsymbol{\gamma}}^2 < \infty.\end{aligned}$$

  By considering the choice $f \equiv 1$, it follows that the above inequality is sharp such that the operator norm of $\mathrm{APP}_{d,2}$ is given by

$$\|\mathrm{APP}_{d,2}\| = 1.$$

- For $p = \infty$, we have for all $f \in \mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$ that

$$\begin{aligned}\|\mathrm{APP}_{d,\infty}(f)\|_{L_\infty} = \|f\|_{L_\infty} &= \sup_{\boldsymbol{x}\in[0,1]^d} |f(\boldsymbol{x})| = \sup_{\boldsymbol{x}\in[0,1]^d} \left|\langle f, K_{d,\alpha,\boldsymbol{\gamma}}(\cdot,\boldsymbol{x})\rangle_{d,\alpha,\boldsymbol{\gamma}}\right| \\ &\le \|f\|_{d,\alpha,\boldsymbol{\gamma}} \sup_{\boldsymbol{x}\in[0,1]^d} \|K_{d,\alpha,\boldsymbol{\gamma}}(\cdot,\boldsymbol{x})\|_{d,\alpha,\boldsymbol{\gamma}} \\ &= \|f\|_{d,\alpha,\boldsymbol{\gamma}} \sup_{\boldsymbol{x}\in[0,1]^d} \sqrt{K_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{x},\boldsymbol{x})} \\ &= \|f\|_{d,\alpha,\boldsymbol{\gamma}} \left(\sum_{\boldsymbol{h}\in\mathbb{Z}^d} r_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{h})\right)^{1/2} \\ &= \|f\|_{d,\alpha,\boldsymbol{\gamma}} \left(\prod_{j=1}^{d} \left(1 + 2\zeta(\alpha)\gamma_j\right)\right)^{1/2} < \infty.\end{aligned}$$

By considering the choice $f = K_{d,\alpha,\boldsymbol{\gamma}}(\cdot, \boldsymbol{x})$, it follows that the above inequality is sharp such that the operator norm of $\mathrm{APP}_{d,\infty}$ is given by

$$\|\mathrm{APP}_{d,\infty}\| = \left(\prod_{j=1}^{d} \left(1 + 2\zeta(\alpha)\gamma_j\right) \right)^{1/2}.$$

2.3 The Worst-Case Setting

The worst-case error of an algorithm $A_{n,d}$ as in (1) is defined as

$$e(A_{n,d}, \mathrm{APP}_{d,p}) := \sup_{\substack{f \in \mathcal{H}_{d,\alpha,\boldsymbol{\gamma}} \\ \|f\|_{d,\alpha,\boldsymbol{\gamma}} \le 1}} \|\mathrm{APP}_{d,p}(f) - A_{n,d}(f)\|_{L_p},$$

and the nth minimal worst-case error with respect to the information class Λ is given by

$$e(n, \mathrm{APP}_{d,p}, \Lambda) := \inf_{A_{n,d} \in \Lambda} e(A_{n,d}, \mathrm{APP}_{d,p}),$$

where the infimum is extended over all linear algorithms of the form (1) with information from the class Λ. In the case $p = \infty$ the essential supremum is used in the calculation of $\|\mathrm{APP}_{d,\infty}(f) - A_{n,d}(f)\|_{L_\infty}$.

The initial error, i.e., the error obtained by approximating f by zero, equals

$$\begin{aligned} e(0, \mathrm{APP}_{d,p}) &= \sup_{\substack{f \in \mathcal{H}_{d,\alpha,\boldsymbol{\gamma}} \\ \|f\|_{d,\alpha,\boldsymbol{\gamma}} \le 1}} \|\mathrm{APP}_{d,p}(f)\|_{L_p} \\ &= \|\mathrm{APP}_{d,p}\| = \begin{cases} 1 & \text{if } p = 2, \\ \left(\prod_{j=1}^{d} \left(1 + 2\zeta(\alpha)\gamma_j\right) \right)^{1/2} & \text{if } p = \infty. \end{cases} \end{aligned}$$

Note that for $p = \infty$ the initial error $e(0, \mathrm{APP}_{d,\infty})$ may be exponential in d if it is not properly normalized. In the following analysis, we will therefore consider the normalized as well as the absolute error criterion.

We are interested in how the approximation error of algorithms $A_{n,d}$ depends on the number n of information evaluations used and how it depends on the problem dimension d. To this end, we define the so-called information complexity as

$$n(\varepsilon, \mathrm{APP}_{d,p}, \Lambda) := \min\{n \in \mathbb{N}_0 : e(n, \mathrm{APP}_{d,p}, \Lambda) \le \varepsilon \, \mathrm{CRI}_{d,p}\}$$

with $\varepsilon \in (0, 1)$ and $d \in \mathbb{N}$, and where either $\mathrm{CRI}_{d,p} = 1$ for the absolute error criterion (we then write $n_{\mathrm{abs}}(\varepsilon, \mathrm{APP}_{d,p}, \Lambda)$) and $\mathrm{CRI}_{d,p} = e(0, \mathrm{APP}_{d,p}) = \|\mathrm{APP}_{d,p}\|$ for the normalized error criterion (then, we write $n_{\mathrm{norm}}(\varepsilon, \mathrm{APP}_{d,p}, \Lambda)$).

2.4 Useful Relations

In the case of L_2-approximation we have $e(0, \mathrm{APP}_{d,2}) = 1$ and hence the absolute and the normalized error criteria coincide. This means that

$$n_{\mathrm{norm}}(\varepsilon, \mathrm{APP}_{d,2}, \Lambda) = n_{\mathrm{abs}}(\varepsilon, \mathrm{APP}_{d,2}, \Lambda)$$

and we just write $n(\varepsilon, \mathrm{APP}_{d,2}, \Lambda)$ for $\Lambda \in \{\Lambda^{\mathrm{all}}, \Lambda^{\mathrm{std}}\}$.

In the case of L_∞-approximation the situation is different, since $e(0, \mathrm{APP}_{d,\infty}) > 1$. Hence we only have

$$n_{\mathrm{norm}}(\varepsilon, \mathrm{APP}_{d,\infty}, \Lambda) \le n_{\mathrm{abs}}(\varepsilon, \mathrm{APP}_{d,\infty}, \Lambda) \quad \text{for} \quad \Lambda \in \{\Lambda^{\mathrm{all}}, \Lambda^{\mathrm{std}}\}. \tag{2}$$

Furthermore, it is well known, see, e.g., [3], that L_2-approximation is not harder than L_∞-approximation for the absolute error criterion, which means that for $\Lambda \in \{\Lambda^{\mathrm{all}}, \Lambda^{\mathrm{std}}\}$ we have

$$n(\varepsilon, \mathrm{APP}_{d,2}, \Lambda) \le n_{\mathrm{abs}}(\varepsilon, \mathrm{APP}_{d,\infty}, \Lambda).$$

Thus, necessary conditions for tractability of L_2-approximation in the weighted space $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$ are also necessary conditions for tractability of L_∞-approximation in $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$ for the absolute error criterion.

For the information class Λ^{all}, L_p-approximation for $p \in \{2, \infty\}$ can be fully characterized in terms of the eigenvalues of the self-adjoint, compact operator

$$W_d := \mathrm{APP}_{d,2}^* \mathrm{APP}_{d,2} : \mathcal{H}_{d,\alpha,\boldsymbol{\gamma}} \to \mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}.$$

The following well-known lemma (see, e.g., [10, p. 215]) provides information on the eigenpairs of the operator W_d.

Lemma 1 *The eigenpairs of the operator W_d are $(r_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{k}), e_{\boldsymbol{k}})$ with $\boldsymbol{k} \in \mathbb{Z}^d$, where for $\boldsymbol{k} \in \mathbb{Z}^d$ we set*

$$e_{\boldsymbol{k}}(\boldsymbol{x}) = e_{\boldsymbol{k},\alpha,\boldsymbol{\gamma}}(\boldsymbol{x}) := \sqrt{r_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{k})}\, \exp(2\pi\, \mathrm{i}\, \boldsymbol{k} \cdot \boldsymbol{x}), \quad \text{for } \boldsymbol{x} \in [0,1]^d.$$

Furthermore, denote the ordered eigenvalues of W_d by $(\lambda_{d,k})_{k \in \mathbb{N}}$, where

$$\lambda_{d,1} \ge \lambda_{d,2} \ge \lambda_{d,3} \ge \cdots.$$

Note that $\lambda_{d,1} = 1$, since $r_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{0}) = 1$ and $\gamma_j \le 1$ for all $j \in \mathbb{N}$.

We then have the following relations (see, for example, [10, 13] for $p = 2$ and [5, Theorem 2] for $p = \infty$) for the nth minimal error with respect to Λ^{all},

$$e(n, \mathrm{APP}_{d,p}, \Lambda^{\mathrm{all}}) = \begin{cases} \lambda_{d,n+1}^{1/2} & \text{if } p = 2, \\ \left(\sum_{k=n+1}^{\infty} \lambda_{d,k}\right)^{1/2} & \text{if } p = \infty. \end{cases}$$

Consequently,

$$n(\varepsilon, \mathrm{APP}_{d,2}, \Lambda^{\mathrm{all}}) = \min\left\{n : \lambda_{d,n+1} \le \varepsilon^2\right\}$$

for $p = 2$, and

$$n(\varepsilon, \mathrm{APP}_{d,\infty}, \Lambda^{\mathrm{all}}) = \min\left\{n : \sum_{k=n+1}^{\infty} \lambda_{d,k} \le \varepsilon^2 \,\mathrm{CRI}_{d,\infty}^2\right\} \tag{3}$$

for $p = \infty$.

2.5 Relations to the Average-Case Setting

Note that (3) is exactly the same as the information complexity for L_2-approximation in the average-case setting for certain spaces (see [12, p. 190] for a general introduction to the average-case setting). Indeed, following the outline in [4], assume that we are given a sequence of spaces $\mathcal{F}_d$, $d \in \mathbb{N}$, and study the operator $\widetilde{\mathrm{APP}}_{d,2} : \mathcal{F}_d \to L_2([0,1]^d)$ with $\widetilde{\mathrm{APP}}_{d,2}(f) = f$ for $f \in \mathcal{F}_d$. Furthermore, we assume that $\mathcal{F}_d$ is equipped with a Gaussian probability measure μ_d, which has mean zero and a covariance function that coincides with the reproducing kernel of the Korobov space $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$, with all parameters as above. I.e.,

$$\int_{\mathcal{F}_d} f(\boldsymbol{x}) f(\boldsymbol{y}) \mu_d(\mathrm{d}f) = K_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{x}, \boldsymbol{y}) \quad \text{for all } \boldsymbol{x}, \boldsymbol{y} \in [0,1]^d.$$

Again, it is of interest to study approximation of $\widetilde{\mathrm{APP}}_{d,2}$ by linear algorithms $A_{n,d}$ of the form (1). The *average-case error* of such an algorithm $A_{n,d}$ is given by

$$e^{\mathrm{avg}}(A_{n,d}, \widetilde{\mathrm{APP}}_{d,2}) := \left(\int_{\mathcal{F}_d} \left\|\widetilde{\mathrm{APP}}_{d,2}(f) - A_{n,d}(f)\right\|_{L_2([0,1]^d)}^2 \mu_d(\mathrm{d}f)\right)^2,$$

and the initial error by

$$e^{\mathrm{avg}}(0, \widetilde{\mathrm{APP}}_{d,2}) := \left(\int_{\mathcal{F}_d} \left\|\widetilde{\mathrm{APP}}_{d,2}(f)\right\|_{L_2([0,1]^d)}^2 \mu_d(\mathrm{d}f)\right)^2.$$

We can also define the nth minimal average-case error of L_2-approximation in $\mathcal{F}_d$ for an information class Λ by

$$e(n, \widetilde{\mathrm{APP}}_{d,2}, \Lambda) := \inf_{A_{n,d} \in \Lambda} e^{\mathrm{avg}}(A_{n,d}, \widetilde{\mathrm{APP}}_{d,2}).$$

Now define, for any Borel set G in $L_2([0,1]^d)$, the inverse image under $\widetilde{\mathrm{APP}}_{d,2}$ by $\widetilde{\mathrm{APP}}_{d,2}^{-1}(G) := \{f \in \mathcal{F}_d \ : \ \mathrm{APP}_{d,2}(f) \in G\}$ and let $\nu_d := \mu_d \circ \widetilde{\mathrm{APP}}_{d,2}^{-1}$. Then, ν_d is a Gaussian measure on $L_2([0,1]^d)$, again with mean zero, and a covariance operator C_{ν_d} given by

$$(C_{\nu_d} f)(\boldsymbol{x}) = \int_{[0,1]^d} K_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{x}, \boldsymbol{y}) f(\boldsymbol{y}) \mathrm{d}\boldsymbol{y} \quad \text{for all } \boldsymbol{x} \in [0,1]^d.$$

For more detailed information we refer to [4] and the references therein.

Using the notation just introduced, there are several relations to be observed between the worst-case setting and the average-case setting. Indeed, it is known that the eigenvalues of the covariance operator C_{ν_d} coincide with the eigenvalues $(\lambda_{d,k})_{k \in \mathbb{N}}$ of the operator W_d introduced above. Furthermore, by making use of the relation between the covariance function of μ_d and the kernel $K_{d,\alpha,\boldsymbol{\gamma}}$, it can easily be shown that

$$e^{\mathrm{avg}}(0, \widetilde{\mathrm{APP}}_{d,2}) = \left(\sum_{\boldsymbol{k} \in \mathbb{Z}_d} r_{d,\alpha,\boldsymbol{\gamma}}(\boldsymbol{k}) \right)^{1/2} = \left(\sum_{k=1}^{\infty} \lambda_{d,k} \right)^{1/2}.$$

Hence the initial error of average-case L_2-approximation in $\mathcal{F}_d$ is exactly the same as the initial error of worst-case L_∞-approximation in $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$. What is more, if one allows information from Λ^{all}, we have

$$e(n, \widetilde{\mathrm{APP}}_{d,2}, \Lambda^{\mathrm{all}}) = \left(\sum_{k=n+1}^{\infty} \lambda_{d,k} \right)^{1/2}$$

for the nth minimal error, i.e., the nth minimal error of average-case L_2-approximation in $\mathcal{F}_d$ equals the nth minimal error of worst-case L_∞-approximation in $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$. For the derivation of these results and further details, we refer to [13, Chap. 6], see also [10].

These observations (which have been pointed out in the literature before) imply that the results on L_∞-approximation in $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$ presented here can also be interpreted as results on average-case L_2-approximation in $\mathcal{F}_d$. Indeed some of the theorems presented on L_∞-approximation below recover some of the results in [4] and the references therein, formulated for the average-case setting there.

2.6 Notions of Tractability

An important goal of tractability theory is to analyze which problems suffer from the curse of dimensionality, i.e., whether there exist $C, \delta > 0$ such that $n(\varepsilon, \mathrm{APP}_{d,p}, \Lambda) \geq C(1+\delta)^d$ for infinitely many $d \in \mathbb{N}$, and which do not. In the latter case it is then an important task to classify the growth rate of the information complexity with respect to the dimension d tending to infinity ($d \to \infty$) and the error threshold ε tending to zero ($\varepsilon \to 0$). Different growth rates are characterized by means of various notions of tractability which are given in the following definition.

Definition 1 Consider the approximation problem $\mathrm{APP}_p = (\mathrm{APP}_{d,p})_{d\geq 1}$ for the information class Λ. We say that for this problem we have:

(a) Strong polynomial tractability (SPT) if there exist non-negative numbers τ, C such that

$$n(\varepsilon, \mathrm{APP}_{d,p}, \Lambda) \leq C\, \varepsilon^{-\tau} \quad \text{for all } d \in \mathbb{N} \text{ and all } \varepsilon \in (0,1). \tag{4}$$

The infimum of all exponents $\tau \geq 0$ such that (4) holds for some $C \geq 0$ is called the exponent of strong polynomial tractability and is denoted by $\tau^*(\Lambda)$.

(b) Polynomial tractability (PT) if there exist non-negative numbers τ, σ, C such that

$$n(\varepsilon, \mathrm{APP}_{d,p}, \Lambda) \leq C\, \varepsilon^{-\tau} d^{\sigma} \quad \text{for all } d \in \mathbb{N} \text{ and all } \varepsilon \in (0,1).$$

(c) Quasi-polynomial tractability (QPT) if there exist non-negative numbers t, C such that

$$n(\varepsilon, \mathrm{APP}_{d,p}, \Lambda) \leq C\, \exp(t\,(1+\ln d)(1+\ln \varepsilon^{-1})) \quad \text{for all } d \in \mathbb{N} \text{ and all } \varepsilon \in (0,1). \tag{5}$$

The infimum of all exponents $t \geq 0$ such that (5) holds for some $C \geq 0$ is called the exponent of quasi-polynomial tractability and is denoted by $t^*(\Lambda)$.

(d) Weak tractability (WT) if

$$\lim_{d+\varepsilon^{-1}\to\infty} \frac{\ln n(\varepsilon, \mathrm{APP}_{d,p}, \Lambda)}{d+\varepsilon^{-1}} = 0.$$

(e) (σ, τ)-weak tractability ((σ, τ)-WT) for positive numbers σ, τ if

$$\lim_{d+\varepsilon^{-1}\to\infty} \frac{\ln n(\varepsilon, \mathrm{APP}_{d,p}, \Lambda)}{d^{\sigma}+\varepsilon^{-\tau}} = 0.$$

(f) Uniform weak tractability (UWT) if (σ, τ)-weak tractability holds for all $\sigma, \tau \in (0, 1]$.

We obviously have the following hierarchy of tractability notions:

$$\text{SPT} \Rightarrow \text{PT} \Rightarrow \text{QPT} \Rightarrow \text{UWT} \Rightarrow (\sigma,\tau)\text{-WT}, \quad \text{for any choice of } (\sigma,\tau) \in (0,1]^2.$$

Furthermore, WT coincides with (σ,τ)-WT for $(\sigma,\tau) = (1,1)$.

It is now our aim to find necessary and sufficient conditions on the weights $\boldsymbol{\gamma}$ of the Korobov space $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$ (for fixed smoothness parameter α) under which the approximation problem is tractable. The characterization of the applicable tractability classes will be done with respect to decay conditions on the weight sequence $\boldsymbol{\gamma} = (\gamma_j)_{j\ge 1}$. To this end, we introduce the following notation.

- The infimum of the sequence $\boldsymbol{\gamma}$ is denoted by $\boldsymbol{\gamma}_I := \inf_{j\ge 1} \gamma_j$.
- The sum exponent $s_{\boldsymbol{\gamma}}$ is defined as

$$s_{\boldsymbol{\gamma}} := \inf\left\{\kappa > 0 \,:\, \sum_{j=1}^{\infty} \gamma_j^{\kappa} < \infty\right\}.$$

- The exponent $t_{\boldsymbol{\gamma}}$ is defined as

$$t_{\boldsymbol{\gamma}} := \inf\left\{\kappa > 0 \,:\, \limsup_{d\to\infty} \frac{1}{\ln(d+1)} \sum_{j=1}^{d} \gamma_j^{\kappa} < \infty\right\}.$$

- The exponent $u_{\boldsymbol{\gamma},\sigma}$, for $\sigma > 0$, is defined as

$$u_{\boldsymbol{\gamma},\sigma} := \inf\left\{\kappa > 0 \,:\, \lim_{d\to\infty} \frac{1}{d^{\sigma}} \sum_{j=1}^{d} \gamma_j^{\kappa} = 0\right\}.$$

In the definitions of $s_{\boldsymbol{\gamma}}$, $t_{\boldsymbol{\gamma}}$, and $u_{\boldsymbol{\gamma},\sigma}$ we use the convention that $\inf \emptyset = \infty$.

3 The Results for APP_2

A complete overview of necessary and sufficient conditions for tractability of L_2-approximation in the weighted Korobov space has recently been published in [2].

Theorem 1 *Consider the approximation problem* $\text{APP}_2 = (\text{APP}_{d,2})_{d\ge 1}$ *for the weighted Korobov spaces* $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$, $d \in \mathbb{N}$, *and for the information class* Λ^{all} *and let* $\alpha > 1$. *Then we have the following results.*

1. *(Cf. [14]) Strong polynomial tractability for the class* Λ^{all} *holds if and only if* $s_{\boldsymbol{\gamma}} < \infty$. *In this case the exponent of strong polynomial tractability is*

$$\tau^*(\Lambda^{\mathrm{all}}) = 2\max\left(s_{\boldsymbol{\gamma}}, \frac{1}{\alpha}\right).$$

2. *(Cf. [14]) Strong polynomial tractability and polynomial tractability for the class* Λ^{all} *are equivalent.*
3. *Quasi-polynomial tractability, uniform weak tractability, and weak tractability for the class* Λ^{all} *are equivalent and hold if and only if* $\boldsymbol{\gamma}_1 < 1$.
4. *If we have quasi-polynomial tractability, then the exponent of quasi-polynomial tractability satisfies*

$$t^*(\Lambda^{\mathrm{all}}) = 2\max\left(\frac{1}{\alpha}, \frac{1}{\ln \boldsymbol{\gamma}_1^{-1}}\right).$$

   In particular, if $\boldsymbol{\gamma}_1 = 0$, *we set* $(\ln \boldsymbol{\gamma}_1^{-1})^{-1} := 0$ *and we have that* $t^*(\Lambda^{\mathrm{all}}) = \frac{2}{\alpha}$.
5. *For* $\sigma > 1$, (σ, τ)*-weak tractability for the class* Λ^{all} *holds for all weights* $1 \geq \gamma_1 \geq \gamma_2 \geq \cdots > 0$.

Theorem 2 *Consider multivariate approximation* $\mathrm{APP}_2 = (\mathrm{APP}_{d,2})_{d\geq 1}$ *for the weighted Korobov spaces* $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$, $d \in \mathbb{N}$, *and for the information class* Λ^{std} *and* $\alpha > 1$. *Then we have the following results.*

1. *(Cf. [9]) Strong polynomial tractability for the class* Λ^{std} *holds if and only if*

$$\sum_{j=1}^{\infty} \gamma_j < \infty,$$

   which implies $s_{\boldsymbol{\gamma}} \leq 1$. *In this case the exponent of strong polynomial tractability satisfies*

$$\tau^*(\Lambda^{\mathrm{std}}) = 2\max\left(s_{\boldsymbol{\gamma}}, \frac{1}{\alpha}\right).$$

2. *(Cf. [9]) Polynomial tractability for the class* Λ^{std} *holds if and only if*

$$\limsup_{d\to\infty} \frac{1}{\ln(d+1)} \sum_{j=1}^{d} \gamma_j < \infty.$$

3. *Polynomial and quasi-polynomial tractability for the class* Λ^{std} *are equivalent.*
4. *Weak tractability for the class* Λ^{std} *holds if and only if*

$$\lim_{d\to\infty} \frac{1}{d} \sum_{j=1}^{d} \gamma_j = 0.$$

5. *For $\sigma \in (0,1]$, (σ,τ)-weak tractability for the class Λ^{std} holds if and only if*

$$\lim_{d\to\infty} \frac{1}{d^\sigma} \sum_{j=1}^{d} \gamma_j = 0.$$

For $\sigma > 1$, (σ,τ)-weak tractability for the class Λ^{std} holds for all weights $1 \geq \gamma_1 \geq \gamma_2 \geq \cdots > 0$.

6. *Uniform weak tractability for the class Λ^{std} holds if and only if*

$$\lim_{d\to\infty} \frac{1}{d^\sigma} \sum_{j=1}^{d} \gamma_j = 0 \quad \text{for all } \sigma \in (0,1].$$

Theorems 1 and 2 imply that in the case of L_2-approximation no open questions remain, at least for the currently most common tractability classes.

Remark 1 For the information class Λ^{std} also the precise form of the algorithms leading to the respective notions of tractability is of interest. However, it would be beyond the scope of this work to provide comprehensive comments on this issue. For the case of (strong) polynomial tractability we refer to the paper [9] for further information. The proof for the weak tractability notions is based on a general relation between tractability for the classes Λ^{all} and Λ^{std}; see [2] as well as [12, Theorem 26.11].

4 The Results for APP_∞

We have the following result for L_∞-approximation in the space $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$.

Theorem 3 *Consider multivariate approximation $\mathrm{APP}_\infty = (\mathrm{APP}_{d,\infty})_{d\geq 1}$ for the weighted Korobov spaces $\mathcal{H}_{d,\alpha,\boldsymbol{\gamma}}$, $d \in \mathbb{N}$, and for the information classes Λ^{all} and Λ^{std} for the normalized and absolute error criterion and $\alpha > 1$. Then we have the following results.*

1. *(Cf. [5] for Λ^{all} and [6] for Λ^{std}) The approximation problem is strongly polynomially tractable if and only if $s_{\boldsymbol{\gamma}} < 1$. If this holds, then for any $\tau \in (\max(1/\alpha, s_{\boldsymbol{\gamma}}), 1)$ we have*

$$e(n, \mathrm{APP}_{d,\infty}, \Lambda^{\mathrm{all}}) = \mathcal{O}\left(n^{-(1-\tau)/(2\tau)}\right) \quad \text{and}$$
$$e(n, \mathrm{APP}_{d,\infty}, \Lambda^{\mathrm{std}}) = \mathcal{O}\left(n^{-(1-\tau)/(2\tau(1+\tau))}\right),$$

where in both cases the implied factor is independent of n and d.

2. *(Cf. [5] for $\Lambda^{\rm all}$ and [6] for $\Lambda^{\rm std}$) The approximation problem is polynomially tractable if and only if $t_{\boldsymbol{\gamma}} < 1$. If this holds, then for any $\tau \in (\max(1/\alpha, t_{\boldsymbol{\gamma}}), 1)$ and any $\delta > 0$ we have*

$$e(n, \mathrm{APP}_{d,\infty}, \Lambda^{\rm all}) = \mathcal{O}\left(n^{-(1-\tau)/(2\tau)} d^{\delta+\zeta(\alpha\tau)t_{\boldsymbol{\gamma}}/\tau}\right) \quad \text{and}$$
$$e(n, \mathrm{APP}_{d,\infty}, \Lambda^{\rm std}) = \mathcal{O}\left(n^{-(1-\tau)/(2\tau(1+\tau))} d^{\delta+\zeta(\alpha\tau)t_{\boldsymbol{\gamma}}/\tau}\right),$$

   where in both cases the implied factor is independent of n and d.
3. *A necessary condition for quasi-polynomial tractability is*

$$\limsup_{d\to\infty} \frac{1}{\ln(d+1)} \sum_{j=1}^{d} \gamma_j < \infty,$$

   which implies $t_{\boldsymbol{\gamma}} \le 1$.
4. *A necessary condition for weak tractability is*

$$\lim_{d\to\infty} \frac{1}{d} \sum_{j=1}^{d} \gamma_j = 0,$$

   which implies $u_{\boldsymbol{\gamma},1} \le 1$, and a sufficient condition for weak tractability is $u_{\boldsymbol{\gamma},1} < 1$.
5. *A necessary condition for (σ, τ)-weak tractability for $\sigma \in (0, 1]$ is*

$$\lim_{d\to\infty} \frac{1}{d^{\sigma}} \sum_{j=1}^{d} \gamma_j = 0,$$

   which implies $u_{\boldsymbol{\gamma},\sigma} \le 1$, and a sufficient condition for (σ, τ)-weak tractability is $u_{\boldsymbol{\gamma},\sigma} < 1$.

   For $\sigma > 1$, (σ, τ)-weak tractability holds for all weights $1 \ge \gamma_1 \ge \gamma_2 \ge \cdots > 0$.
6. *A necessary condition for uniform weak tractability is*

$$\lim_{d\to\infty} \frac{1}{d^{\sigma}} \sum_{j=1}^{d} \gamma_j = 0 \quad \text{for all } \sigma \in (0, 1],$$

   which implies $u_{\boldsymbol{\gamma},\sigma} \le 1$ for all $\sigma \in (0, 1]$, and a sufficient condition for uniform weak tractability is

$$u_{\boldsymbol{\gamma},\sigma} < 1 \quad \text{for all } \sigma \in (0, 1].$$

Remark 2 Some remarks on Theorem 3 are in order.

1. So far we only have a necessary condition for QPT, which is

$$\limsup_{d\to\infty} \frac{1}{\ln(d+1)} \sum_{j=1}^{d} \gamma_j < \infty, \tag{6}$$

   and which in turn implies $t_{\boldsymbol{\gamma}} \le 1$. However, this condition is very close to the "if and only if"-condition for PT, which is $t_{\boldsymbol{\gamma}} < 1$. It is an interesting question whether (6) is already strong enough to imply QPT or whether $t_{\boldsymbol{\gamma}} < 1$ is really necessary. The latter case would imply that PT and QPT are equivalent.
2. The necessary and sufficient conditions for the notions of weak tractability in Items 4–6 are very tight, although not matching exactly. How to close these gaps is another interesting problem. Regarding Item 4, we also refer to [10, Sect. 6.3], where a corresponding result for L_2-approximation in the average-case setting is shown, and this is—as pointed out in our remarks above—equivalent to our result for L_∞-approximation in the worst-case setting. There, the same gap is observed, but the authors of [10] point out that at least for general weights the condition $\lim_{d\to\infty} \frac{1}{d} \sum_{j=1}^{d} \gamma_j = 0$ is not sufficient for weak tractability. Whether a similar observation also holds for the special case of product weights, which are considered in the present paper, remains open.
3. Again, for the information class Λ^{std}, it is natural to ask for the precise form of the algorithms leading to the respective notions of tractability. Similarly to our remark above, we point out that it would go beyond the scope of this work to provide comprehensive comments regarding this question. For the case of (strong) polynomial tractability we refer to [6] for more detailed information. For the other cases the tractability results are based on a spline algorithm as indicated in the proof below; see [15] for further information.

Proof of Theorem 3 Proofs of the results on (strong) polynomial tractability in Items 1 and 2 can be found in [5, Theorem 11] for the class Λ^{all} and in [6, Theorem 11] for Λ^{std}.

Now we consider QPT. From (3) and the fact that $\lambda_{d,k} \le 1$ for all $k \in \mathbb{N}$, we have for $n = n_{\mathrm{norm}}(\varepsilon, \mathrm{APP}_{d,\infty}, \Lambda^{\mathrm{all}})$ that

$$\sum_{k=1}^{\infty} \lambda_{d,k} - n \le \sum_{k=n+1}^{\infty} \lambda_{d,k} \le \varepsilon^2 \sum_{k=1}^{\infty} \lambda_{d,k}.$$

Hence,

$$n \ge (1-\varepsilon^2) \sum_{k=1}^{\infty} \lambda_{d,k} = (1-\varepsilon^2) \prod_{j=1}^{d} \left(1 + 2\zeta(\alpha)\gamma_j\right). \tag{7}$$

Assume that we have QPT for L_∞-approximation for $\Lambda^{\rm all}$ and the normalized error criterion. Then there exist positive t and C such that

$$C\mathrm{e}^{t(1+\ln d)(1+\ln \varepsilon^{-1})} \geq n_{\rm norm}(\varepsilon, \mathrm{APP}_{d,\infty}, \Lambda^{\rm all}) \geq (1-\varepsilon^2) \prod_{j=1}^{d} \big(1+2\zeta(\alpha)\gamma_j\big)$$

for all $d \in \mathbb{N}$ and all $\varepsilon \in (0,1)$.

Fixing $\varepsilon \in (0,1)$, e.g., choosing $\varepsilon = \mathrm{e}^{-1}$ and taking the logarithm implies the condition

$$\ln C + 2t(1+\ln d) \geq \ln\left(\frac{\mathrm{e}^2-1}{\mathrm{e}^2}\right) + \sum_{j=1}^{d} \ln(1+2\zeta(\alpha)\gamma_j)$$

for all $d \in \mathbb{N}$. This implies $\lim_{j\to\infty} \gamma_j = 0$. Since $\frac{\ln(1+x)}{x} \to 1$ for $x \to 0$, this then implies

$$\limsup_{d\to\infty} \frac{1}{\ln(d+1)} \sum_{j=1}^{d} \gamma_j < \infty. \tag{8}$$

Thus we have shown that (8) is a necessary condition for QPT for $\Lambda^{\rm all}$ and the normalized error criterion. Since QPT for $\Lambda^{\rm std}$ implies QPT for $\Lambda^{\rm all}$, we find that (8) is also a necessary condition for QPT for $\Lambda^{\rm std}$ and the normalized error criterion.

Assume that we have QPT for L_∞-approximation for $\Lambda \in \{\Lambda^{\rm all}, \Lambda^{\rm std}\}$ and the absolute error criterion. Then, according to (2), we have QPT for L_∞-approximation for Λ and the normalized error criterion, and hence (8) holds. Thus the proof of Item 3 is complete.

We now discuss (σ,τ)-WT and first consider the necessary conditions. Assume that we have (σ,τ)-WT for $\sigma \in (0,1]$ for L_∞-approximation for $\Lambda^{\rm all}$ and the normalized error criterion. Then, according to (7),

$$\begin{aligned} 0 &= \lim_{d+\varepsilon^{-1}\to\infty} \frac{\ln n_{\rm norm}(\varepsilon, \mathrm{APP}_{d,\infty}, \Lambda^{\rm all})}{d^\sigma + \varepsilon^{-\tau}} \\ &\geq \lim_{d+\varepsilon^{-1}\to\infty} \left(\frac{\ln(1-\varepsilon^2)}{d^\sigma+\varepsilon^{-\tau}} + \frac{\sum_{j=1}^{d} \ln(1+2\zeta(\alpha)\gamma_j)}{d^\sigma+\varepsilon^{-\tau}} \right). \end{aligned}$$

For fixed $\varepsilon \in (0,1)$ this implies

$$\lim_{d\to\infty} \frac{1}{d^\sigma} \sum_{j=1}^{d} \ln(1+2\zeta(\alpha)\gamma_j) = 0,$$

which in turn implies that

$$\lim_{d\to\infty} \frac{1}{d^\sigma} \sum_{j=1}^{d} \gamma_j = 0. \tag{9}$$

So (9) is a necessary condition for (σ, τ)-WT for $\sigma \in (0, 1]$ for $\Lambda^{\rm all}$ and the normalized error criterion. In the same way as for QPT we see that (9) is a necessary condition for (σ, τ)-WT for $\sigma \in (0, 1]$ for $\Lambda \in \{\Lambda^{\rm all}, \Lambda^{\rm std}\}$ and the normalized and the absolute error criterion. Note that (9) implies $u_{\boldsymbol{\gamma},\sigma} \le 1$. This finishes the proof of the necessary conditions in Items 4–6.

Next, we discuss sufficient conditions for (σ, τ)-WT. In [15] Zeng, Kritzer, and Hickernell constructed a spline algorithm $A_{n,d}^{\rm spline}$ based on lattice rules with a prime number n of nodes, for which for arbitrary $\lambda \in (1/2, \alpha/2)$

$$e(A_{n,d}^{\rm spline}, {\rm APP}_{d,\infty}) \le \frac{\sqrt{2}}{n^{\lambda(2\lambda-1)/(4\lambda-1)}} \prod_{j=1}^{d} \left(1 + 2^{2\alpha+1} \gamma_j^{1/(2\lambda)} \zeta\left(\frac{\alpha}{2\lambda}\right)\right)^{2\lambda}. \tag{10}$$

Assume that $u_{\boldsymbol{\gamma},\sigma} < 1$. Then there exists a $\lambda \in (1/2, \alpha/2)$ such that

$$\lim_{d\to\infty} \frac{1}{d^\sigma} \sum_{j=1}^{d} \gamma_j^{1/(2\lambda)} = 0. \tag{11}$$

We show that (11) implies (σ, τ)-WT for the class $\Lambda^{\rm std}$ and the absolute error criterion (and therefore also for the class $\Lambda^{\rm all}$ and, because of (2), the same holds true for the normalized error criterion).

Let

$$M := \left\lceil \left(\frac{\sqrt{2}}{\varepsilon} \prod_{j=1}^{d} \left(1 + 2^{2\alpha+1} \gamma_j^{1/(2\lambda)} \zeta\left(\frac{\alpha}{2\lambda}\right)\right)^{2\lambda} \right)^{(4\lambda-1)/(\lambda(2\lambda-1))} \right\rceil$$

and let n be the smallest prime number that is greater than or equal to M. Note that then, according to Bertrand's postulate, $n \in [M, 2M]$. Hence, according to (10) we have

$$e(n, {\rm APP}_{d,\infty}, \Lambda^{\rm std}) \le \varepsilon,$$

and therefore

$$n(\varepsilon, {\rm APP}_{d,\infty}, \Lambda^{\rm std}) \le n \le 2M \le 4 \left(\frac{\sqrt{2}}{\varepsilon} \prod_{j=1}^{d} \left(1 + 2^{2\alpha+1} \gamma_j^{1/(2\lambda)} \zeta\left(\frac{\alpha}{2\lambda}\right)\right)^{2\lambda} \right)^{(4\lambda-1)/(\lambda(2\lambda-1))}.$$

Taking the logarithm and using that $\ln(1+x) \leq x$ for $x \geq 0$ yields

$$\ln n(\varepsilon, \mathrm{APP}_{d,\infty}, \Lambda^{\mathrm{std}}) \leq \ln 4 + \frac{4\lambda - 1}{\lambda(2\lambda - 1)} \left[\frac{\ln 2}{2} + \ln \varepsilon^{-1} + 2^{2(\alpha+1)} \lambda \zeta \left(\frac{\alpha}{2\lambda} \right) \sum_{j=1}^{d} \gamma_j^{1/(2\lambda)} \right]$$

and hence

$$\begin{aligned} &\lim_{d+\varepsilon^{-1} \to \infty} \frac{\ln n(\varepsilon, \mathrm{APP}_{d,\infty}, \Lambda^{\mathrm{std}})}{d^{\sigma} + \varepsilon^{-\tau}} \\ &\leq \frac{4\lambda - 1}{\lambda(2\lambda - 1)} \left[\lim_{d+\varepsilon^{-1} \to \infty} \frac{\ln \varepsilon^{-1}}{d^{\sigma} + \varepsilon^{-\tau}} + 2^{2(\alpha+1)} \lambda \zeta \left(\frac{\alpha}{2\lambda} \right) \lim_{d+\varepsilon^{-1} \to \infty} \frac{1}{d^{\sigma} + \varepsilon^{-\tau}} \sum_{j=1}^{d} \gamma_j^{1/(2\lambda)} \right] \\ &= 0, \end{aligned}$$

where we used (11) for the case $\sigma \in (0, 1]$ in the last step. If $\sigma > 1$ then (11) is not required, since $\gamma_j \leq 1$ for all $j \in \mathbb{N}$, and so the limit relation holds anyway. Thus the proof of Items 4–6 is finished. □

Remark 3 Let us briefly comment on the L_p-approximation problem $\mathrm{APP}_p = (\mathrm{APP}_{d,p})_{d \geq 1}$ for $p \in (2, \infty)$ and the absolute error criterion. As for the relation between the minimal errors of L_2- and L_∞-approximation, it can be shown that

$$e(n, \mathrm{APP}_{d,2}, \Lambda) \leq e(n, \mathrm{APP}_{d,p}, \Lambda) \leq e(n, \mathrm{APP}_{d,\infty}, \Lambda) \quad \text{for all } n, d \in \mathbb{N},$$

and

$$n(\varepsilon, \mathrm{APP}_{d,2}, \Lambda) \leq n(\varepsilon, \mathrm{APP}_{d,p}, \Lambda) \leq n(\varepsilon, \mathrm{APP}_{d,\infty}, \Lambda).$$

for all $\varepsilon \in (0, 1)$, and all $d \in \mathbb{N}$.

Therefore, we can conclude that a sufficient condition on the weights for a certain tractability notion for L_∞-approximation is also sufficient for the same tractability notion for the L_p-approximation problem. The other way round, every necessary condition on the weights for a certain tractability notion for L_2-approximation is also necessary for the same tractability notion for the L_p-approximation problem. We summarize the results that are implied by this insight in Table 3 in Sect. 5. However, many of the sufficient and necessary conditions which we obtain in this way are far from matching each other (especially the ones for the class Λ^{all}). Whether a similar observation is also true for the normalized error criterion and for $p \in (2, \infty)$ remains an open question.

5 Overview and Formulation of Open Problems

In Tables 1, 2 and 3 below, we give a concise overview of the known results and conditions for the various tractability notions. Table 1 is concerned with L_2-approximation, Table 2 with L_∞-approximation, and Table 3 with L_p-approximation for $p \in (2, \infty)$.

5.1 Open Problems

While we have a full picture of the characterizations of the currently most common notions of tractability for the L_2-approximation problem, the L_∞-case is only partially solved and several details remain open. In particular, for QPT a sufficient

Table 1 Overview of the conditions for tractability of the L_2-approximation problem APP_2 for product weights satisfying $1 \geq \gamma_1 \geq \gamma_2 \geq \cdots > 0$ (recall that normalized and absolute criterion coincide for APP_2)

	Λ^{all}	Λ^{std}
SPT	$s_{\boldsymbol{\gamma}} < \infty$	$\sum_{j=1}^{\infty} \gamma_j < \infty$
PT	$s_{\boldsymbol{\gamma}} < \infty$	$\limsup_{d\to\infty} \frac{1}{\ln(d+1)} \sum_{j=1}^{d} \gamma_j < \infty$
QPT	$\boldsymbol{\gamma}_I < 1$	$\limsup_{d\to\infty} \frac{1}{\ln(d+1)} \sum_{j=1}^{d} \gamma_j < \infty$
UWT	$\boldsymbol{\gamma}_I < 1$	$\lim_{d\to\infty} \frac{1}{d^\sigma} \sum_{j=1}^{d} \gamma_j = 0 \ \forall \sigma \in (0, 1]$
(σ, τ)-WT, $\sigma \in (0, 1]$	$\boldsymbol{\gamma}_I < 1$	$\lim_{d\to\infty} \frac{1}{d^\sigma} \sum_{j=1}^{d} \gamma_j = 0$
WT	$\boldsymbol{\gamma}_I < 1$	$\lim_{d\to\infty} \frac{1}{d} \sum_{j=1}^{d} \gamma_j = 0$
(σ, τ)-WT, $\sigma > 1$	No extra condition on $\boldsymbol{\gamma}$	No extra condition on $\boldsymbol{\gamma}$

Table 2 Overview of the conditions for tractability of the L_∞-approximation problem APP_∞ for product weights satisfying $1 \geq \gamma_1 \geq \gamma_2 \geq \cdots > 0$ (normalized and absolute criterion)

	Λ^{all} and Λ^{std}
SPT	$s_{\boldsymbol{\gamma}} < 1$
PT	$t_{\boldsymbol{\gamma}} < 1$
QPT	nec.: $\limsup_{d\to\infty} \frac{\sum_{j=1}^{d} \gamma_j}{\ln(d+1)} < \infty$
UWT	$\begin{cases} \text{nec.: } \lim_{d\to\infty} \frac{\sum_{j=1}^{d} \gamma_j}{d^\sigma} = 0 \ \forall \sigma \in (0, 1] \\ \text{suff.: } u_{\boldsymbol{\gamma},\sigma} < 1 \ \forall \sigma \in (0, 1] \end{cases}$
(σ, τ)-WT, $\sigma \in (0, 1]$	$\begin{cases} \text{nec.: } \lim_{d\to\infty} \frac{\sum_{j=1}^{d} \gamma_j}{d^\sigma} = 0 \\ \text{suff.: } u_{\boldsymbol{\gamma},\sigma} < 1 \end{cases}$
WT	$\begin{cases} \text{nec.: } \lim_{d\to\infty} \frac{\sum_{j=1}^{d} \gamma_j}{d} = 0 \\ \text{suff.: } u_{\boldsymbol{\gamma},1} < 1 \end{cases}$
(σ, τ)-WT, $\sigma > 1$	No extra condition on $\boldsymbol{\gamma}$

Table 3 Overview of the conditions for tractability of the L_p-approximation problem APP_p, $p \in (2, \infty)$, for product weights satisfying $1 \geq \gamma_1 \geq \gamma_2 \geq \cdots > 0$ (absolute criterion)

	Λ^{all}	Λ^{std}
SPT	$\begin{cases} \text{nec.: } s_{\boldsymbol{\gamma}} < \infty \\ \text{suff.: } s_{\boldsymbol{\gamma}} < 1 \end{cases}$	$\begin{cases} \text{nec.: } s_{\boldsymbol{\gamma}} \leq 1 \\ \text{suff.: } s_{\boldsymbol{\gamma}} < 1 \end{cases}$
PT	$\begin{cases} \text{nec.: } s_{\boldsymbol{\gamma}} < 1 \\ \text{suff.: } t_{\boldsymbol{\gamma}} < 1 \end{cases}$	$\begin{cases} \text{nec.: } \limsup_{d\to\infty} \frac{\sum_{j=1}^{d} \gamma_j}{\ln(d+1)} < \infty \\ \text{suff.: } t_{\boldsymbol{\gamma}} < 1 \end{cases}$
QPT	$\begin{cases} \text{nec.: } \gamma_I < 1 \\ \text{suff.: ?} \end{cases}$	$\begin{cases} \text{nec.: } \limsup_{d\to\infty} \frac{\sum_{j=1}^{d} \gamma_j}{\ln(d+1)} < \infty \\ \text{suff.: ?} \end{cases}$
UWT	$\begin{cases} \text{nec.: } \gamma_I < 1 \\ \text{suff.: } u_{\boldsymbol{\gamma},\sigma} < 1 \ \forall \sigma \in (0, 1] \end{cases}$	$\begin{cases} \text{nec.: } \lim_{d\to\infty} \frac{\sum_{j=1}^{d} \gamma_j}{d^{\sigma}} = 0 \ \forall \sigma \in (0, 1] \\ \text{suff.: } u_{\boldsymbol{\gamma},\sigma} < 1 \ \forall \sigma \in (0, 1] \end{cases}$
(σ, τ)-WT, $\sigma \in (0, 1]$	$\begin{cases} \text{nec.: } \gamma_I < 1 \\ \text{suff.: } u_{\boldsymbol{\gamma},\sigma} < 1 \end{cases}$	$\begin{cases} \text{nec.: } \lim_{d\to\infty} \frac{\sum_{j=1}^{d} \gamma_j}{d^{\sigma}} = 0 \\ \text{suff.: } u_{\boldsymbol{\gamma},\sigma} < 1 \end{cases}$
WT	$\begin{cases} \text{nec.: } \gamma_I < 1 \\ \text{suff.: } u_{\boldsymbol{\gamma},1} < 1 \end{cases}$	$\begin{cases} \text{nec.: } \lim_{d\to\infty} \frac{\sum_{j=1}^{d} \gamma_j}{d} = 0 \\ \text{suff.: } u_{\boldsymbol{\gamma},1} < 1 \end{cases}$
(σ, τ)-WT, $\sigma > 1$	No extra condition on $\boldsymbol{\gamma}$	No extra condition on $\boldsymbol{\gamma}$

condition is still missing, and for the (σ, τ)-weak tractability notions the necessary and sufficient conditions are tight, but do not match (see Remark 2). These cases remain open for the moment.

Furthermore, also the more general L_p-approximation problem for arbitrary $p \in (2, \infty)$ remains unsolved. While there are some fragmentary results for the absolute error criterion, the case of the normalized criterion is completely open (see Remark 3).

Acknowledgements The authors would like to thank an anonymous referee for helpful comments. The authors are supported by the Austrian Science Fund (FWF), Projects F5506-N26 (Ebert and Kritzer) and F5509-N26 (Pillichshammer), which are parts of the Special Research Program "Quasi-Monte Carlo Methods: Theory and Applications", as well as P34808 (Kritzer).

References

1. Bakhvalov, N.S.: On the optimality of linear methods for operator approximation in convex classes of functions. USSR Comput. Math. Math. Phys. **11**, 244–249 (1971)
2. Ebert, A., Pillichshammer, F.: Tractability of approximation in the weighted Korobov spaces in the worst-case setting–a complete picture. J. Complex. **67**, 101571, 15 pp. (2021)
3. Kritzer, P., Pillichshammer, F., Woźniakowski, H.: L_∞-approximation in Korobov spaces with exponential weights. J. Complex. **41**, 102–125 (2017)
4. Kuo, F.Y., Sloan, I.H., Woźniakowski, H.: Lattice rule algorithms for multivariate approximation in the average case setting. J. Complex. **24**, 283–323 (2008)
5. Kuo, F.Y., Wasilkowski, G.W., Woźniakowski, H.: Multivariate L_∞ approximation in the worst-case setting over reproducing kernel Hilbert spaces. J. Approx. Theory **152**, 135–160 (2008)

6. Kuo, F.Y., Wasilkowski, G.W., Woźniakowski, H.: On the power of standard information for multivariate approximation in the worst-case setting. J. Approx. Theory **158**, 97–125 (2009)
7. Leobacher, G., Pillichshammer, F.: Introduction to Quasi-Monte Carlo Integration and Applications. Compact Textbooks in Mathematics. Birkhäauser/Springer, Cham (2014)
8. Novak, E.: Some results on the complexity of numerical integration. Monte Carlo and Quasi-Monte Carlo Methods, pp. 161–183, Springer Proceedings in Mathematics & Statistics, vol. 163. Springer, Cham (2016)
9. Novak, E., Sloan, I.H., Woźniakowski, H.: Tractability of approximation for weighted Korobov spaces on classical and quantum computers. Found. Comput. Math. **4**(2), 121–156 (2004)
10. Novak, E., Woźniakowski, H.: Tractability of Multivariate Problems. Volume I: Linear Information. EMS, Zurich (2008)
11. Novak, E., Woźniakowski, H.: Tractability of Multivariate Problems. Volume II: Standard Information for Functionals. EMS, Zurich (2010)
12. Novak, E., Woźniakowski, H.: Tractability of Multivariate Problems. Volume III: Standard Information for Operators. EMS, Zurich (2012)
13. Traub, J.F., Wasilkowski, G.W., Woźniakowski, H.: Information-Based Complexity. Academic, New York (1988)
14. Wasilkowski, G.W., Woźniakowski, H.: Weighted tensor product algorithms for linear multivariate problems. J. Complex. **15**(3), 402–447 (1999)
15. Zeng, X., Kritzer, P., Hickernell, F.J.: Spline methods using integration lattices and digital nets. Constr. Approx. **30**, 529–555 (2009)

Rare-Event Simulation via Neural Networks

Lachlan J. Gibson and Dirk P. Kroese

Abstract We present a neural network framework for the simulation of independent random variables from arbitrary distributions. The framework includes two neural networks that are trained simultaneously using a target function, rather than a target dataset. One is a generative model that maps samples from a joint normal distribution to samples in the target space. The second network estimates the probability density of these samples, trained with targets obtained via kernel density estimation. The effectiveness of the approach is illustrated with various examples from rare-event simulation. The generator was able to learn all the 1-dimensional distribution examples well enough to pass the Kolmogorov–Smirnov test. However, estimates of higher-dimensional probability densities were limited by the kernel density estimation. Refining the density estimates of the generated samples is a clear way to improve the accuracy of the method when learning more complex higher dimensional distributions.

Keywords Random variable generation · Neural networks · Rare events · Generative networks · Deep learning · Simulation

1 Introduction

Ever since the inception of electronic computing, there has been a demand for fast and reliable methods to simulate random experiments on a computer. Computations with neural networks are nowadays so commonplace and powerful that the question arises whether such networks can be used to generate independent samples from any complicated probability distribution. This is particularly pertinent in the field of *rare-event simulation*, where sampling from the conditional distribution given that a rare event takes place can be prohibitively difficult.

L. J. Gibson (✉) · D. P. Kroese
School of Mathematics and Physics, The University of Queensland, Brisbane, Australia
e-mail: l.gibson1@uq.edu.au

D. P. Kroese
e-mail: kroese@maths.uq.edu.au

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_8

Neural networks are already used extensively to simulate artificial data, by using so-called *generative models*, and so devising a generator network to simulate independent random variables from a prescribed distribution should be feasible. We introduce network architecture and training procedures to learn to generate samples from a given target distribution that is known up to possibly a normalization constant. A typical instance arises in rare-event simulation, where the objective is to simulate a random object $\boldsymbol{X} \in \mathcal{X}$ from a specified probability density h, conditional upon the occurrence of an event $\{S(\boldsymbol{X}) \geq \gamma\}$, where S is a positive *performance* function and γ a *level* or *rarity* parameter. The conditional pdf of $\boldsymbol{X}$ given $\{S(\boldsymbol{X}) \geq \gamma\}$ is then given by

$$\frac{h(\boldsymbol{x})\mathbb{1}_{\{S(\boldsymbol{x})\geq\gamma\}}}{c}, \quad \boldsymbol{x} \in \mathcal{X}, \tag{1}$$

where the normalization constant $c = \mathbb{E}\mathbb{1}_{\{S(\boldsymbol{x})\geq\gamma\}} = \mathbb{P}(S(\boldsymbol{X}) \geq \gamma)$ is typically unknown. Here, $\mathbb{1}_A$ denotes the indicator random variable of an event A.

Our contribution should be seen as a *pilot study* to provide a neural network framework for independent sampling from an arbitrary target distribution that is known up to possibly a normalization constant. We do this by introducing two networks: a *generator* network and a probability *density* network. The generator network takes standard normal random vectors as input and aims to return (after training) an independent sample from the target distribution. The probability density network provides (after training) an estimate of the probability density corresponding to the generator network. Both networks are trained simultaneously.

Once training is completed, the network output can be combined with importance sampling—for example, to estimate the normalization constant of the target distribution. Although in principle the class of sampling distributions provided by the generator network is parametric, the number of parameters is so large that effectively any pdf can be matched very closely. This is particularly useful when dealing with truncated densities, as in (1).

The advantage of this approach is that by feeding the network independent normals, the output samples are independent as well, even in a rare-event simulation setting. This is in contrast to well-known sampling methods such as Markov chain Monte Carlo and splitting. The implementation in Python is made freely available for experimentation and requires little programming. An example implementation is given in the Appendix.

1.1 Background

Much research has gone into devising fast and reliable random number generators, which lie at the heart of every simulation algorithm. Two of the most popular generators are the Mersenne twister [36] and MRG32k3a [30]. The latter is much easier to implement than the former and passes all known statistical tests (in contrast to the earlier version of the Mersenne twister). Other interesting generators, including

those based on $\mathbb{F}_2$ arithmetic, are given in [29, 32]. The ultimate test suite for random number generator is *TESTU01* by [33].

Many methods for non-uniform random variable generation are given in [12]. A wide range of algorithms for random process simulation, including spatial processes are discussed in [4, 24, 26].

There are many uses for simulation and Monte Carlo methods, ranging from simulation modelling [13, 28], optimization [1, 3, 7], counting [6] to the solving of difficult estimation problems [23]. Typical applications in finance, queueing and reliability can for example be found in [8, 14, 22, 34, 37, 43]. In statistics, the MCMC method is ubiquitous [9, 17].

The area of rare-event simulation has a long history. Two generic techniques for rare-event simulation are *importance sampling* [42] and *splitting* [15, 19, 31]. Both methods aim to increase the probability of rare events.

In splitting, this is achieved by duplicating promising trajectories that could more likely lead to the rare event. In importance sampling the probability distribution under which the process is simulated is changed to make the rare event more likely. In particular, for rare-event probabilities of the form $c = \mathbb{P}(S(\boldsymbol{X}) \geq \gamma)$ in (1), one can, in principle, estimate c via the estimator

$$\hat{c} := \frac{1}{n} \sum_{k=1}^{n} \mathbb{1}_{\{S(\boldsymbol{X}_k) \geq \gamma\}} \frac{h(\boldsymbol{X}_k)}{g(\boldsymbol{X}_k)}, \tag{2}$$

were $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$ is an iid sample from an arbitrary pdf g. In fact, the pdf in (1) gives the optimal importance sampling distribution, yielding a *zero-variance* estimator. Of course, if c is unknown, this pdf cannot be evaluated. Using an inappropriate IS density g can lead to estimates that differ from c by orders of magnitude. Knowing the form of the zero-variance pdf is important in devising rare-event simulation techniques with theoretical guarantees of efficiency. Moreover, the zero-variance distribution provides essential insights in the way that rare events happen.

An often-used importance sampling approach is to seek an optimal *exponential change of measure* through a large deviation analysis [10]. This usually requires that the distribution of interest is *light-tailed*. For rare-event simulation with *heavy-tailed* distributions, a different strategy needs to be used; see e.g., [5]. In *adaptive* importance sampling methods, the sampling distribution is interactively updated to lie increasingly closer to the optimal importance sampling distribution within a parametric class of distributions. The *cross-entropy* (CE) method [40] is a well-known example. The method uses the *Kullback–Leibler* divergence [27] to measure the proximity (relative entropy) between two probability densities. The CE method can also be used to optimize complicated multi-modal functions; see, e.g., [7].

The field of deep learning has achieved several significant advances over the last decade. Specifically, the recent growth in computing power and improvement in algorithms has enabled the training of deep neural networks to succeed in a broad range of tasks. This has revolutionized many fields, such as computer vision [44], natural language processing [38], recommender systems [45], fraud detection [11],

and many more. There are many types of neural networks, but one standard class is the multilayer perceptron (MLP). The MLP is a type of feed-forward network comprised of a sequence of layers, with the first being the input, the last being the output and all intermediate layers being 'hidden'. The node values at layer i form a vector $\boldsymbol{x}_i$ which serves as an input to compute the subsequent layer via

$$\boldsymbol{x}_{i+1} = \sigma_i(\boldsymbol{W}_i \boldsymbol{x}_i + \boldsymbol{b}_i), \tag{3}$$

where $\boldsymbol{W}_i$ is a matrix of weights, $\boldsymbol{b}_i$ is a vector of biases and σ_i is the so-called activation function at layer i. Choosing non-linear activation functions introduces non-linearity into the network which can improve its ability to learn complicated functions. Some typical activation function choices include the logistic function and the rectified linear unit (ReLU) which are applied elementwise. The MLP is a composite function of all these layers, generally containing a large number of learnable parameters (the weights and biases). The number of layers is the network 'depth' and the number of nodes per layer is the network 'width'. The width of each hidden layer, the depth of the network and the activation function at each layer form the network architecture, and are typically chosen prior to training. A short introduction to the mathematics behind multilayer neural networks may be found in [25, Chap. 9].

According to universal approximation theorems, MLPs are arbitrary function approximators [18], in that any well-behaved function can be approximated to any degree of precision given sufficient width or depth (hence parameters) of the network and appropriate activation functions. Therefore, these networks have huge potential for modeling, provided appropriate parameters can be identified. A common method for training MLPs to learn these optimal parameters is through gradient descent, where a loss function that measures performance can be minimized by iteratively updating the network parameters by small steps based on their loss function derivatives. The layered structure of MLPs allows these derivatives to be efficiently computed via the chain rule, a process called backpropagation [41]. Modern software packages such as TensorFlow [2] and PyTorch [39] make these algorithms accessible by computing the derivatives automatically and efficiently performing parallel processing on hardware, such as GPUs.

Deep neural networks can also form *generative* models, such as generative adversarial networks (GANs) [16] and variational autoencoders (VAEs) [21] which have proven to be very successful. These kinds of models aim to generate samples following the same distribution as a provided data-set. However, such data-sets in the context of rare-event simulation are not typically accessible. Therefore, we wish to apply deep learning methods to the field of rare-event simulation to train a generative model to be able to generate rare events without using a training dataset. Additionally, we want to provide a basic Python software implementation.

The rest of the chapter is organized as follows. Section 2 explains the details of the new method. Section 3 has results from several examples, and Sect. 4 concludes the work and outlines potential avenues for further research.

2 Rare-Event Deep Learning

We introduce a procedure to train a generator deep neural network to learn how to generate iid samples from a target distribution, where the corresponding probability density is known up to a constant factor. This also allows the learning of conditional densities, so that the generator can sample rare events directly. This section outlines the details of the networks and training procedures that are involved.

2.1 Networks and Loss Functions

The generator network, G, is a feedforward neural network, such as a multilayer perceptron (MLP), that takes as input a d-dimensional standard normal random vector $\boldsymbol{Z}$; we write this as $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$. The (many) parameters of the network are gathered in a vector $\boldsymbol{u}$. The generator network G maps each "noise" input $\boldsymbol{Z}$ to a target variable

$$\boldsymbol{X} := G(\boldsymbol{Z}; \boldsymbol{u}). \tag{4}$$

The aim is to construct G such that $\boldsymbol{X}$ has the target probability density $f(\boldsymbol{x})/c$ at the point $\boldsymbol{x}$, where $f(\boldsymbol{x})$ can be evaluated, but the normalization constant, c, may be unknown. In the setting of (1), $f(\boldsymbol{x}) = h(\boldsymbol{x})\mathbb{1}_{\{S(\boldsymbol{x})\geq\gamma\}}$. In other words, the goal is to learn appropriate values of $\boldsymbol{u}$ to make the density of sampling $\boldsymbol{x}$ from $G(\boldsymbol{Z}; \boldsymbol{u})$ as close to $f(\boldsymbol{x})/c$ as possible. This can be achieved by minimizing the Kullback–Leibler (KL) divergence from the target density to the generator density. Let $g(\boldsymbol{x}; \boldsymbol{u})$ represent the probability density of sampling $\boldsymbol{x}$ from the generator with parameters $\boldsymbol{u}$. The KL divergence is given by

$$\mathcal{D}(g, f/c) = \mathbb{E} \ln \frac{g(\boldsymbol{X}; \boldsymbol{u})}{f(\boldsymbol{X})} + \ln c, \tag{5}$$

where $\mathbb{E}$ represents the expectation with respect to the generator distribution with parameters $\boldsymbol{u}$. Note that the KL divergence is not symmetrical and that $\mathcal{D}(g, f)$ is different from $\mathcal{D}(f, g)$ used in the CE method. This change is motivated by a key difference in how the parameters are updated during training. Unlike the CE method which fixes generated points while updating parameters to optimize the density function, our method fixes an estimate of the density function while updating parameters to optimize the generated points. Specifically, points sampled in regions where $g(\boldsymbol{x}; \boldsymbol{u}) > f(\boldsymbol{x})/c$ need to be 'pushed' towards regions where $g(\boldsymbol{x}; \boldsymbol{u}) < f(\boldsymbol{x})/c$, which can be achieved by minimizing the specified generator loss while keeping g fixed during the parameter updates.

Evaluating the probability density function $g(\boldsymbol{x}; \boldsymbol{u})$ is not always straightforward, and for this reason we introduce a second network, $D(\boldsymbol{x}; \boldsymbol{v})$, with parameters $\boldsymbol{v}$, to estimate the probability density of the generator network. This gives rise to the *generator loss*, defined as

$$L_G(\boldsymbol{u}, \boldsymbol{v}) := \mathcal{D}(D, f/c) - \ln c = \mathbb{E} \ln \frac{D(\boldsymbol{X}; \boldsymbol{v})}{f(\boldsymbol{X})}, \tag{6}$$

which drops the $\ln c$ term from the KL divergence, since it does not depend on any network parameters.

To ensure that the probability density network represents a good estimate of the densities produced by the generator network for fixed parameters $\boldsymbol{u}$, the parameters $\boldsymbol{v}$ of the density network are chosen to minimize the expected squared error between the network log-density and the true log-density. So we define the *density loss* as

$$L_D(\boldsymbol{u}, \boldsymbol{v}) := \mathbb{E} \left(\ln D(\boldsymbol{X}; \boldsymbol{v}) - \ln g(\boldsymbol{X}; \boldsymbol{u})\right)^2. \tag{7}$$

Obviously, $L_D(\boldsymbol{u}, \boldsymbol{v})$ is minimal (and equal to 0) when $D(\boldsymbol{x}; \boldsymbol{v}) = g(\boldsymbol{x}; \boldsymbol{u})$. In that case, the smallest value that $L_G(\boldsymbol{u}, \boldsymbol{v})$ can take is $-\ln c$, when $g(\boldsymbol{x}; \boldsymbol{u}) = f(\boldsymbol{x})/c$. Therefore, training the generator and density networks to learn the desired distribution involves solving the following minimization program:

$$\min_{\boldsymbol{u}} L_G\big(\boldsymbol{u}, \operatorname*{argmin}_{\boldsymbol{v}} L_D(\boldsymbol{u}, \boldsymbol{v})\big). \tag{8}$$

To achieve this, we utilize a form of stochastic gradient descent whereby batches $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n\}$ of n samples from the generator network are used to estimate the loss functions. Specifically, $L_G(\boldsymbol{u}, \boldsymbol{v})$ is estimated by

$$\widehat{L}_G(\boldsymbol{u}, \boldsymbol{v}) = \frac{1}{n} \sum_{k=1}^{n} \ln \frac{D(\boldsymbol{X}_k; \boldsymbol{v})}{f(\boldsymbol{X}_k)}, \qquad \boldsymbol{X}_k = G(\boldsymbol{Z}_k; \boldsymbol{u}), \qquad \boldsymbol{Z}_k \sim \mathcal{N}(0, \mathbf{I}). \tag{9}$$

and $L_D(\boldsymbol{u}, \boldsymbol{v})$ is estimated by

$$\widehat{L}_D(\boldsymbol{u}, \boldsymbol{v}) = \frac{1}{n} \sum_{k=1}^{n} \left(\ln D(\boldsymbol{X}_k; \boldsymbol{v}) - \ln \widehat{g}(\boldsymbol{X}_k; \boldsymbol{u})\right)^2, \quad \boldsymbol{X}_k = G(\boldsymbol{Z}_k; \boldsymbol{u}), \quad \boldsymbol{Z}_k \sim \mathcal{N}(0, \mathbf{I}), \tag{10}$$

where $\hat{g}$ is an estimate of g obtained from the batch. In principle, $\hat{g}$ could be used to train the generator network directly without requiring the use of the density network. However, there are practical benefits to including the density network. Specifically, using a density network can ensure the density estimate is compatible with gradient descent optimization, by being a well behaved function whose derivatives can be stably computed using automatic differentiation. Additionally, the density network estimates the probability density of individual points independently, without needing a large sample. This improves stability during training, since the density estimates have less variance, and provides a relatively compact model of the generator probability density function after the training has finished.

2.2 Kernel Density Estimation

Computing the density loss of a sample via Eq. (10) requires an estimate of the generator probability density function. We estimate $g(\boldsymbol{x}; \boldsymbol{u})$ via the *kernel density estimator*

$$\widehat{g}(\boldsymbol{x}; \boldsymbol{u}) := \frac{1}{n} \sum_{k=1}^{n} \varphi(\boldsymbol{x}, \boldsymbol{X}_k; \Sigma), \tag{11}$$

where the "kernel" is a normalized multivariate Gaussian centered at $\boldsymbol{X}_k$ with covariance matrix $\boldsymbol{\Sigma}$:

$$\varphi(\boldsymbol{x}, \boldsymbol{X}_k; \Sigma) := \det(2\pi\Sigma)^{-\frac{1}{2}} \mathrm{e}^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{X}_k)^{\intercal}\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{X}_k)}. \tag{12}$$

In general, $\boldsymbol{\Sigma}$ can be any symmetric positive definite matrix of correct dimensionality, but for simplicity we choose $\Sigma = \sigma^2 \boldsymbol{I}$, where σ represents the bandwidth and $\boldsymbol{I}$ is the identity matrix. An 'optimal' bandwidth for a given sample can be chosen via least-squares cross-validation to minimize the *integrated squared error*, given by

$$\begin{aligned} \mathrm{ISE} &= \int [\widehat{g}(\boldsymbol{x}; \boldsymbol{u}) - g(\boldsymbol{x}; \boldsymbol{u})]^2 \, \mathrm{d}\boldsymbol{x}, \\ &= \int \widehat{g}(\boldsymbol{x}; \boldsymbol{u})^2 \mathrm{d}\boldsymbol{x} - 2 \int g(\boldsymbol{x}; \boldsymbol{u}) \widehat{g}(\boldsymbol{x}; \boldsymbol{u}) \mathrm{d}\boldsymbol{x} + \int g(\boldsymbol{x}; \boldsymbol{u})^2 \mathrm{d}\boldsymbol{x}. \end{aligned} \tag{13}$$

The third term in Eq. (13) is independent of the kernel, so the bandwidth can be chosen to minimize just the first two terms. Substituting the Gaussian kernel from Eq. (12) into Eq. (11), the first term can be evaluated as

$$\int \widehat{g}(\boldsymbol{x}; \boldsymbol{u})^2 \mathrm{d}\boldsymbol{x} = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \varphi(\boldsymbol{X}_i, \boldsymbol{X}_j; 2\Sigma), \tag{14}$$

while the second term can be estimated using the unbiased cross-validation estimator

$$\int g(\boldsymbol{x}; \boldsymbol{u}) \widehat{g}(\boldsymbol{x}; \boldsymbol{u}) \mathrm{d}\boldsymbol{x} \approx \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j \neq i} \varphi(\boldsymbol{X}_i, \boldsymbol{X}_j; \Sigma), \tag{15}$$

which computes the sample mean of the kernel density estimator at each sample point that omits that point. Therefore, for a given sample, the bandwidth is chosen to minimize

$$s(\Sigma) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \varphi(\boldsymbol{X}_i, \boldsymbol{X}_j; 2\Sigma) - 2 \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j \neq i} \varphi(\boldsymbol{X}_i, \boldsymbol{X}_j; \Sigma). \tag{16}$$

We use a form of gradient descent to find $\text{argmin}_\sigma s\left(\sigma^2 \boldsymbol{I}\right)$, where $\boldsymbol{I}$ is the identity matrix of the same size as Σ.

2.3 Training Procedure

The objective shown in Eq. (8) involves the simultaneous minimization of both the generator and density losses. To achieve this, we use a form of stochastic gradient descent which alternates between updating the parameters of the generator network and the density network. The training procedure, outlined in Algorithm 1, aims to train the density network at a faster rate than the generator network, to ensure it represents a reasonable estimate of the generator probability density function. The networks are initialized with random parameters and the kernel bandwidth is optimized by minimizing Eq. (16) via a form of gradient descent. Then, the density network is trained for several steps using Algorithm 3. Then for a fixed number of epochs, the generator network is updated for one or more steps via Algorithm 2 and the density network is updated for one or more steps. The number of steps the density network is updated is typically larger than the number of steps the generator network is updated, to ensure the density network can adapt to the changes in the generator network.

Figure 1 depicts a flowchart representation of the network training steps.

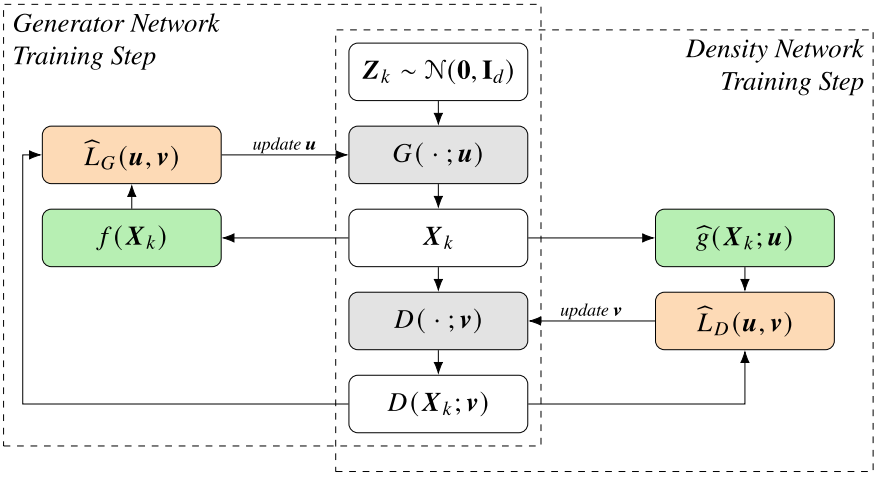


Fig. 1 A flowchart representation of the network training steps. The estimate of the optimal bandwidth is updated at a fixed epoch interval. The left side of the figure shows a generator network training step as outlined in Algorithm 2 and the right shows a density network training step as outlined in Algorithm 3. The networks are shown in gray, the losses shown in orange and training targets shown in green. The network inputs and outputs are shown in white

Algorithm 1: Concurrent training of both generator and density networks

Update optimal kernel bandwidth estimate;
Train density network for several steps using Algorithm 3;
Evaluate networks;
for *several epochs* **do**
    Train generator network for some steps using Algorithm 2;
    if *epoch is at a pre-specified interval* **then**
        Update optimal kernel bandwidth estimate;
    Train density network for several steps using Algorithm 3;
    Evaluate networks;

Algorithm 2: Training generator network

for *several steps* **do**
    Sample n samples of noise $\boldsymbol{Z}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$;
    Generator network maps noise to target space $\boldsymbol{X}_k \leftarrow G(\boldsymbol{Z}_k; \boldsymbol{u})$;
    Density network estimates the probability density of the samples $D(\boldsymbol{X}_k; \boldsymbol{v})$;
    Compute the target at each sample $f(\boldsymbol{X}_k)$;
    Estimate generator loss $\widehat{L}_G(\boldsymbol{u}, \boldsymbol{v})$;
    Adjust parameters $\boldsymbol{u}$ via standard gradient descent methods;

Algorithm 3: Training density network

for *several steps* **do**
    Sample n samples of noise $\boldsymbol{Z}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$;
    Generator network maps noise to target space $\boldsymbol{X}_k \leftarrow G(\boldsymbol{Z}_k; \boldsymbol{u})$;
    Density network estimates the probability density of the samples $D(\boldsymbol{X}_k; \boldsymbol{v})$;
    Estimate the true densities at each sample using KDE $\widehat{g}(\boldsymbol{X}_k; \boldsymbol{u})$;
    Estimate density loss $\widehat{L}_D(\boldsymbol{u}, \boldsymbol{v})$;
    Adjust parameters $\boldsymbol{v}$ via standard gradient descent methods;

2.4 Rare-Event Distribution

In the rare-event setting, the generator aims to generate samples from the conditional distribution with probability density given by (1), with $f(\boldsymbol{x}) = h(\boldsymbol{x})\mathbb{1}_{\{S(\boldsymbol{x})\geq\gamma\}}$. However, the generator loss, calculated by Eq. (6), is only defined when $f(\boldsymbol{x}) > 0$. So, we set the target to $f(\boldsymbol{x}) = h(\boldsymbol{x})\rho(\boldsymbol{x})$ instead, where $\rho(\boldsymbol{x})$, defined by

$$\rho(\boldsymbol{x}) := \mathrm{e}^{-\alpha(\gamma - S(\boldsymbol{x}))\mathbb{1}_{\{S(\boldsymbol{x})<\gamma\}}} \approx \mathbb{1}_{\{S(\boldsymbol{x})\geq\gamma\}}, \tag{17}$$

penalizes non-rare-events exponentially with $\gamma - S(\boldsymbol{x})$. This also facilitates automatic differentiation in the neural networks. The parameter $\alpha > 0$ determines the

strength of the penalty and should be large enough to maintain the approximation. Evaluating the generator loss with this target yields

$$L_G(\boldsymbol{u}, \boldsymbol{v}) = \mathbb{E} \ln \frac{D(\boldsymbol{X}; \boldsymbol{v})}{h(\boldsymbol{X})} + \alpha \mathbb{E}(\gamma - S(\boldsymbol{X})) \mathbb{1}_{\{S(\boldsymbol{X}) < \gamma\}}. \tag{18}$$

The first term is minimized to 0 when $D(\boldsymbol{x}; \boldsymbol{v}) = h(\boldsymbol{x})$, pushing the generator to learn the unconditional target distribution. The additional second term is minimized to 0 when all generated samples are rare-events with $S(\boldsymbol{X}) \geq \gamma$, pushing the generator to sample more rare-events.

3 Experimental Results

We apply our pilot framework to a few simple examples, to demonstrate its strengths and weaknesses and highlight aspects that need further development. In each example the network architecture of the generator and density networks are mostly the same. The generator takes 8-dimensional standard normal random vector inputs. Both the generator and density networks have three hidden layers of sizes 16, 32 and 64 nodes. The hidden layers all use the rectified linear unit (ReLU) activation function, while the density network output layer uses the exponential activation function ensuring a strictly positive density estimate.[1] The output layer activation function of the generator network is the identity function when the target space included all reals, and is an exponential function when the target space was non-negative. The Adam gradient descent algorithm [20] was used to train both networks in each example, with a weight decay parameter of 0.0001. The Python code for each example is available at https://github.com/LachlanGibson/NNRareEvent with one example included in the Appendix.

3.1 Learning Normal Distributions

In the first two examples, the generator aims to learn 1-dimensional distributions with normalized targets. These are to demonstrate the ability of the generator to learn simple distributions. The first example is the standard normal distribution with a target of

$$f(x) = h(x) = \frac{1}{\sqrt{2\pi}} \mathrm{e}^{-\frac{x^2}{2}}, \quad x \in \mathbb{R}. \tag{19}$$

[1] Actually, in the code the density network output activation function is the identity function, but the output is interpreted as the log-density.

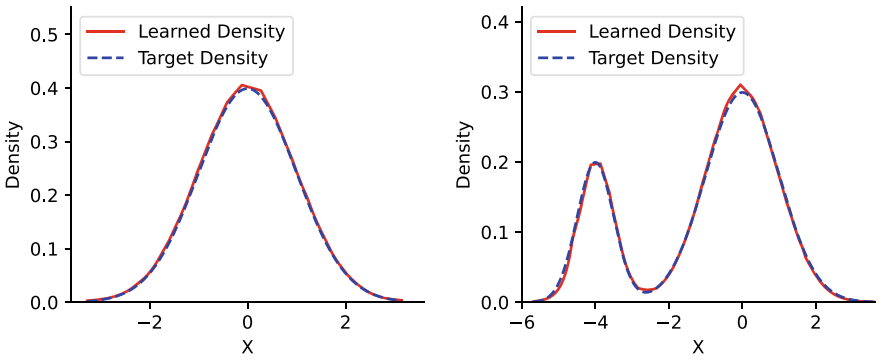


Fig. 2 A comparison of learned and target probability densities. Learned probability densities, as estimated by the density networks, are represented by solid red lines. The dashed blue lines represent the target probability densities of Eqs. (19) and (20). On the left, the generator learns a standard normal distribution. On the right, the generator learns a mixture of two normal distributions

The second example is a bimodal distribution with a target of

$$f(x) = h(x) = \frac{1}{4\sqrt{2\pi 0.5^2}} e^{-\frac{(x+4)^2}{2\times 0.5^2}} + \frac{3}{4\sqrt{2\pi}} e^{-\frac{x^2}{2}}, \quad x \in \mathbb{R}. \tag{20}$$

One of the key limitations of gradient descent based optimization is the possibility of convergence to a local minimum of the loss function, rather than the desired global minimum. In the context of multi-modal distributions with peaks separated by large distances of low density, the generator might only learn a subset of the distribution, never exploring one or more modes of the distribution.

In both examples the generator is able to learn the target distributions sufficiently well to pass the one sample Kolmogorov–Smirnov test. Generating two sets of samples each of sizes 1000 and 10000 the p-values in the first example are about 0.46 and 0.19, which are too high to reject the null hypothesis that the generator distribution equals the target distribution. Likewise, the p-values are about 0.61 and 0.25 in the second example. Figure 2 compares the estimated density learned by the density network with the target densities of Eqs. (19) and (20).

3.2 Normal Distribution Rare-Events

In this third example, the generator aims to learn a truncated standard normal distribution; in particular, the normal distribution conditional on the event $\{S(X) = X \geq \gamma\}$. Including the penalty factor from Eq. (17), the (unnormalized) target is thus given by

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2} - \alpha(\gamma - x))\mathbb{1}_{\{x<\gamma\}}}, \quad x \in \mathbb{R}. \tag{21}$$

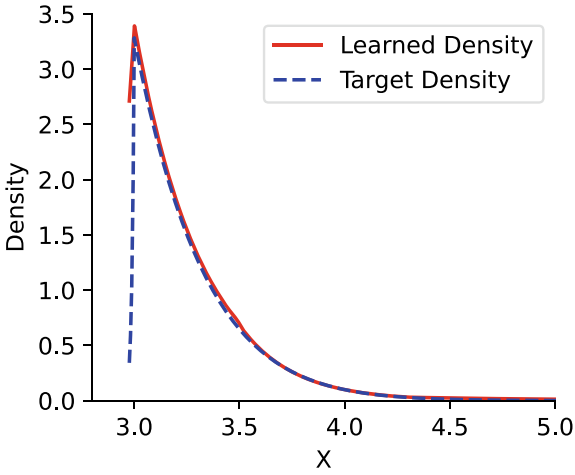


Fig. 3 A comparison of learned and target probability densities of a truncated normal distribution. The target distribution is a normal distribution $\mathcal{N}(0, 1^2)$ with $S(x) = x$ and rarity $\gamma = 3$. The red line is the density learnt by the probability density network. The dashed blue line represents the target density given by renormalizing (21)

For $\gamma = 3$, the probability of sampling a value $X \geq \gamma$ from a normal distribution is about 0.0013499, which also forms the normalization constant of (21) in the limit as $\alpha \to \infty$. Choosing $\alpha = 100$, the generator passed the one sample Kolmogorov–Smirnov test after training with a p-value of 0.15 with sample of size 1000, but failed with sample of size 10000 with a p-value of 0.0021. Figure 3 compares the estimated density learned by the density network with the target density given by renormalizing (21). Based on Eq. (2) an estimator of the normalization constant can be obtained via importance sampling, and depends on how well the density network estimates the generator probability density function. This estimator is given by

$$\hat{c} := \frac{1}{n} \sum_{k=1}^{n} \frac{f(\boldsymbol{X}_k)}{D(\boldsymbol{X}_k; \boldsymbol{v})}. \tag{22}$$

With a sample size of 1000 points, the constant is estimated as 0.0013030 with standard error of 0.0000027. This is within about 3.5% error of the target value 0.0013499.

3.3 Learning Sum of Exponential Distributions

In this final example, the generator aims to learn a 2-dimensional distribution. The target distribution is the joint distribution of two independent exponential random variables, conditional on the event $\{S(X, Y) \geq \gamma\}$, where $S(x, y) = x + y$. Including the penalty factor from Eq. (17), the target is given by

$$f(x, y) = \mathrm{e}^{-x-y} \mathrm{e}^{-\alpha(\gamma - x - y)\mathbb{1}_{\{x+y<\gamma\}}}, \quad x, y \in \mathbb{R}^+. \tag{23}$$

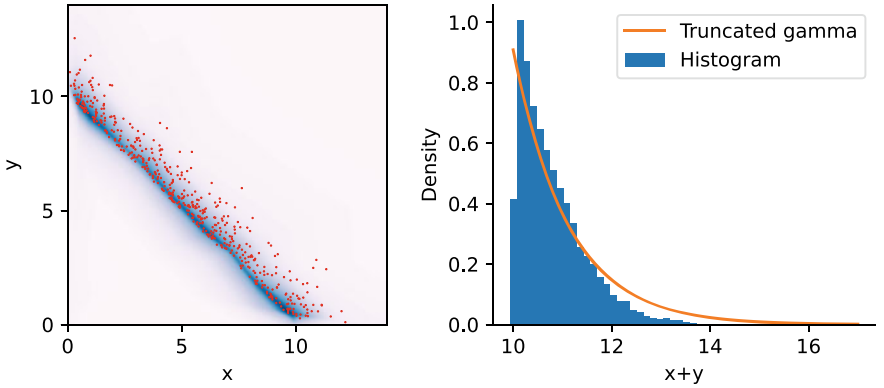


Fig. 4 Sum of two exponential distributions. Threshold at $x + y = 10$. On the left plot, colour represents probability density estimated by the density network. Red dots are a sample of 500 points generated by the generator. On the right, a histogram with 50 bins of 10000 samples of $x + y$ is compared to a truncated gamma distribution

In this case the generator successfully learned to generate rare-events, but with a distribution only roughly approximating the target distribution. Figure 4 includes a scatter plot of 500 points sampled from the trained generator network, and a histogram of $x + y$ compared to the renormalized target truncated gamma distribution. While the scatter plot indicates the points are sampled fairly uniformly along $x + y$ contours, the histogram shows that the generator network undersamples larger values of $x + y$. Furthermore, the importance sampling estimate computed using densities estimated by the density network, 0.0004014 with standard error 0.0000020, is about 20% off the correct value of about 0.0004994.

We believe the network training was limited by the kernel density estimation. Errors learned by the density network could warp the learned generator distribution and could account for much of the 20% discrepancy in the normalization constant estimation, since this calculation relies on those density estimates. For further evidence, we estimated the normalization constant using density estimates found via kernel density estimation (rather than the density network) for a range of kernel bandwidths. Figure 5 illustrates how the estimated normalization constant is sensitive to the kernel bandwidth. For bandwidths that predict values close to the correct normalization constant, the predicted values vary by about 0.7% for every 1% change in bandwidth.

The kernel density estimation would likely be improved by increasing the sample size, but this is quickly limited by hardware, especially for higher dimensional samples. Different kernel functions or more complex kernel covariances could lead to more accurate estimates. Another alternative, would be to replace the density network, with a network that learns the CDF of the generator using the empirical CDF of generated samples, such as discussed in [35]. But in this case, an estimate of the generator probability density is less accessible, requiring the derivative of the outputs of the CDF network with respect to its inputs.

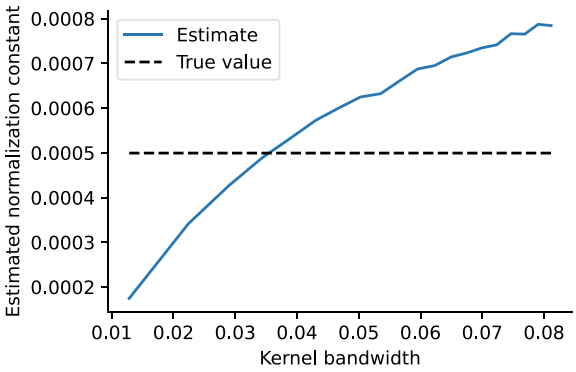


Fig. 5 The dependence on kernel bandwidth σ when estimating the normalization constant. The solid blue line represents the normalization constant estimated by importance sampling with generator probability densities estimated by kernel density estimation. 10000 samples are used to compute each estimate. The dashed black line shows the true rare-event probability of about 0.000499399

The Python code used to run these four experiments is available at https://github.com/LachlanGibson/NNRareEvent. It includes a core file 'rare_event.py' of classes and functions that make experiments relatively simple to code. To demonstrate, the code used for the last sum of exponentials example is shown in the Appendix. The general layout of such a script is as follows. Firstly, the relevant classes and functions are imported from 'rare_event.py'. Next, an instance of the *Generator* class is created, which contains both the generator and density networks. Functions to represent the log-target, $\ln f(\boldsymbol{x})$, and performance, $S(\boldsymbol{x})$, are defined. The inputs of these functions should be torch tensors of shape (# samples, # dimensions) while the outputs have shape (# samples, 1). Next, the kernel bandwidth is optimized using the *optimise_t* method, before the density network is trained using the *train_system* function. The *train_system* function is used again with different arguments to train both the generator network and the density network. This function will save checkpoints of the *Generator* instance depending on the *save_interval* argument. Once training has finished, plots can be generated and statistical tests performed.

4 Conclusions and Further Research

We have introduced a neural network framework for independent sampling from an arbitrary target distribution that is known up to possibly a normalization constant. Our pilot study that examined four examples shows that the framework has potential for rare-event simulation, especially when sampling a 1-dimensional distribution. Quantifying performance when learning a broad range of distributions, such as disconnected or heavy tailed distributions, is a good avenue of further research. Additionally, limitations in the kernel density estimation begin to appear when learn-

ing higher dimensional distributions, which requires further investigation. Provided this hurdle can be resolved, the method could be applied to much higher dimensional contexts, such as simulating rare-events in stochastic processes.

For example, trajectories of a particle undergoing Brownian motion can be represented by a Gaussian stochastic process $\{\boldsymbol{X}_t, t \in \mathcal{T}\}$ where $\boldsymbol{X}_t$ represents the change in spatial position of the particle at time step t, such that the position at step T is given by

$$\boldsymbol{R}_T = \sum_{t=1}^{T} \boldsymbol{X}_t. \tag{24}$$

The joint probability density of a trajectory is the product of Gaussian densities of each dimension in each time step. A set of rare events could be the set of trajectories that pass within distance $-\gamma$ a particular location $\boldsymbol{L}$ at time T,

$$\{-\|\boldsymbol{R}_T - \boldsymbol{L}\| \geq \gamma\}. \tag{25}$$

In this case, the generator network would learn to generate trajectories conditional on the final or an intermediate location.

Appendix

```
from rare_event import *

# Create instance of the full system. Note the generator outputs
# are forced to be positive by the exp activation function.
h = Generator(
    num_dims=2,                  #generator output dimensionality
    num_input_dims = 8,          #generator input dimensionality
    genHLsizes = [16,32,64],     #generator hidden layer sizes
    pdf_netHLsizes = [16,32,64], #density net hidden layer sizes
    generator_activation = torch.exp, #generator activation func
    name = "exponential_sum_gamma10"  #name of system
    ).to(device)

gamma = 10 # The performance threshold

# The target log-density function (up to a constant).
def exponential(X):
    return -X.sum(1, keepdim = True)

# The sample performance function
def S(X):
    return X.sum(1, keepdim = True)
```

```
# Optimize the kernel bandwidth for samples of size 1000.
# The large learning rate of 0.1 accelerates the calculation.
h.optimise_t(1000, lr = 0.1)

# Train the system, only updating the density network 1000 steps.
train_system(h,1,exponential,S,gen_steps=0,pdf_steps=1000,
gamma=gamma,save_interval=1,kernel_interval=1,bs=1000)

# Optimize the kernel bandwidth for samples of size 10000.
h.optimise_t(10000, lr = 0.1)

# Train the system for 3000 epochs. At each epoch the generator
# is updated once and the density network is updated 10 times.
train_system(h,3000,exponential,S,gen_steps=1,pdf_steps=10,
gamma=gamma,save_interval=100,kernel_interval=10,bs=10000)

# Lower the learning rate of the generator optimizer from the
# default of 0.001 to 0.0001.
h.generator.optimiser.param_groups[0]["lr"] = 0.0001

# Train the system for another 3000 epochs.
train_system(h,3000,exponential,S,gen_steps=1,pdf_steps=10,
gamma=gamma,save_interval=100,kernel_interval=10,bs=10000)

# Plot a contour plot
h.plot2Dcontour(xrange = [0,14], xn = 1000, levels = 100)
plt.show()
```

References

1. Aarts, E.H.L., Korst, J.H.M.: Simulated Annealing and Boltzmann Machines. Wiley, Chichester (1989)
2. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: large-scale machine learning on heterogeneous systems (2015). https://www.tensorflow.org/. Software available from tensorflow.org
3. Andradóttir, S.: A global search method for discrete stochastic optimization. SIAM J. Optim. **6**, 513–530 (1996)
4. Asmussen, S., Glynn, P.W.: Stochastic Simulation. Springer, New York (2007)
5. Asmussen, S., Kroese, D.P.: Improved algorithms for rare event simulation with heavy tails. Adv. Appl. Probab. **38**(2), 545–558 (2006)
6. Blanchet, J.: Importance sampling and efficient counting for binary contingency tables. Ann. Appl. Probab. **19**, 949–982 (2009)
7. Botev, Z.I., Kroese, D.P., Rubinstein, R., L'Ecuyer, P.: The cross-entropy method for optimization. In: V. Govindaraju, C. Rao (eds.) Handbook of Statistics, vol. 31: Machine Learning, pp. 19–34. Elsevier, Chennai (2013)

8. Botev, Z.I., L'Ecuyer, P., Simard, R., Tuffin, B.: Static network reliability estimation under the Marshall–Olkin copula. ACM Trans. Model. Comput. Simul. **26**(2) (2016). https://doi.org/10.1145/2775106
9. Brooks, S., Gelman, A., Jones, G., Meng, X.L.: Handbook of Markov Chain Monte Carlo. CRC Press, Boca Raton (2011)
10. Bucklew, J.A.: Introduction to Rare Event Simulation. Springer, New York (2004)
11. Chalapathy, R., Chawla, S.: Deep learning for anomaly detection: a survey (2019)
12. Devroye, L.: Non-Uniform Random Variate Generation. Springer, New York (1986)
13. Fishman, G.S.: Discrete Event Simulation: Modeling, Programming, and Analysis. Springer, New York (2001)
14. Glasserman, P.: Monte Carlo Methods in Financial Engineering. Springer, New York (2004)
15. Glasserman, P., Heidelberger, P., Shahabuddin, P., Zajic, T.: A look at multilevel splitting. In: Niederreiter, H. (ed.) Monte Carlo and Quasi Monte Carlo Methods 1996. Lecture Notes in Statistics, vol. 127, pp. 99–108. Springer, New York (1996)
16. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks. Commun. ACM **63**(11), 139–144 (2020). https://doi.org/10.1145/3422622.
17. Hastings, W.K.: Monte Carlo sampling methods using Markov chains and their applications. Biometrika **57**, 92–109 (1970)
18. Hornik, K.: Approximation capabilities of multilayer feedforward networks. Neural Netw. **4**(2), 251–257 (1991). https://doi.org/10.1016/0893-6080(91)90009-T
19. Kahn, H., Harris, T.E.: Estimation of particle transmission by random sampling. National Bureau of Standards Applied Mathematics Series (1951)
20. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2017)
21. Kingma, D.P., Welling, M.: An introduction to variational autoencoders. Foundations and Trends® in Machine Learning **12**(4), 307–392 (2019). https://doi.org/10.1561/2200000056
22. Kloeden, P.E., Platen, E.: Numerical Solution of Stochastic Differential Equations. Springer, New York (1992). Corrected third printing
23. Kroese, D., Rubinstein, R.Y., Glynn, P.W.: The cross-entropy method for estimation. In: V. Govindaraju, C. Rao (eds.) Handbook of Statistics, vol. 31: Machine Learning, pp. 35–59. Elsevier, Chennai (2013)
24. Kroese, D.P., Botev, Z.I.: Spatial process simulation. In: V. Schmidt (ed.) Lectures on Stochastic Geometry, Spatial Statistics and Random Fields, vol. II: Analysis, Modeling and Simulation of Complex Structures. Springer, Berlin (2014)
25. Kroese, D.P., Botev, Z.I., Taimre, T., Vaisman, R.: Data Science and Machine Learning: Mathematical and Statistical Methods. Chapman and Hall/CRC, Boca Raton (2019)
26. Kroese, D.P., Taimre, T., Botev, Z.I.: Handbook of Monte Carlo Methods. Wiley, New York (2011)
27. Kullback, S.: Information Theory and Statistics. Wiley, New York (1959)
28. Law, A.M., Kelton, W.D.: Simulation Modeling and Analysis, 3rd edn. McGraw-Hill, New York (2000)
29. L'Ecuyer, P.: Random numbers for simulation. Commun. ACM **33**(10), 85–97 (1990)
30. L'Ecuyer, P.: Good parameters and implementations for combined multiple recursive random number generators. Oper. Res. **47**(1), 159–164 (1999)
31. L'Ecuyer, P., Demers, V., Tuffin, B.: Splitting for rare-event simulation. ACM Trans. Model. Comput. Simul. (TOMACS) **17**(2), 1–44 (2007)
32. L'Ecuyer, P., Panneton, F.: $\mathbb{F}_2$-linear random number generators. In: Alexopoulos, C., Goldsman, D., Wilson, J.R. (eds.) Advancing the Frontiers of Simulation: A Festschrift in Honor of George Samuel Fishman, pp. 175–200. Springer, New York (2009)
33. L'Ecuyer, P., Simard, R.: TestU01: a C library for empirical testing of random number generators. ACM Trans. Math. Softw. **33**(4) (2007). Article 22
34. Lieber, D., Rubinstein, R.Y., Elmakis, D.: Quick estimation of rare events in stochastic networks. IEEE Trans. Reliab. **46**, 254–265 (1997)

35. Magdon-Ismail, M., Atiya, A.: Density estimation and random variate generation using multilayer networks. IEEE Trans. Neural Netw. **13**(3), 497–520 (2002). https://doi.org/10.1109/TNN.2002.1000120
36. Matsumoto, M., Nishimura, T.: Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans. Model. Comput. Simul. **8**(1), 3–30 (1998)
37. McLeish, D.L.: Monte Carlo Simulation and Finance. Wiley, New York (2005)
38. Otter, D.W., Medina, J.R., Kalita, J.K.: A survey of the usages of deep learning for natural language processing. IEEE Trans. Neural Netw. Learn. Syst. **32**(2), 604–624 (2021). https://doi.org/10.1109/TNNLS.2020.2979670
39. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019). http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
40. Rubinstein, R.Y., Kroese, D.P.: The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte Carlo Simulation and Machine Learning. Springer, New York (2004)
41. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature **323**(6088), 533–536 (1986). https://doi.org/10.1038/323533a0.
42. Siegmund, D.: Importance sampling in the Monte Carlo study of sequential tests. Ann. Stat. **4**, 673–684 (1976)
43. Tsoucas, P.: Rare events in series of queues. J. Appl. Probab. **29**, 168–175 (1992)
44. Voulodimos, A., Doulamis, N., Doulamis, A., Protopapadakis, E.: Deep learning for computer vision: a brief review. Comput. Intell. Neurosci. **2018**, 7068349 (2018). https://doi.org/10.1155/2018/7068349.
45. Zhang, S., Yao, L., Sun, A., Tay, Y.: Deep learning based recommender system: a survey and new perspectives. ACM Comput. Surv. **52**(1) (2019). https://doi.org/10.1145/3285029

Preintegration is Not Smoothing When Monotonicity Fails

Alexander D. Gilbert, Frances Y. Kuo, and Ian H. Sloan

Abstract Preintegration is a technique for high-dimensional integration over the d-dimensional Euclidean space, which is designed to reduce an integral whose integrand contains kinks or jumps to a $(d-1)$-dimensional integral of a smooth function. The resulting smoothness allows efficient evaluation of the $(d-1)$-dimensional integral by a Quasi-Monte Carlo or Sparse Grid method. The technique is similar to conditional sampling in statistical contexts, but the intention is different: in conditional sampling the aim is usually to reduce the variance, rather than to achieve smoothness. Preintegration involves an initial integration with respect to one well chosen real-valued variable. Griebel, Kuo, Sloan [*Math. Comp. 82 (2013), 383–400*] and Griewank, Kuo, Leövey, Sloan [*J. Comput. Appl. Maths. 344 (2018), 259–274*] showed that the resulting $(d-1)$-dimensional integrand is indeed smooth under appropriate conditions, including a key assumption—that the smooth function underlying the kink or jump is *strictly monotone* with respect to the chosen special variable when all other variables are held fixed. The question addressed in this paper is whether this monotonicity property with respect to one well chosen variable is necessary. We show here that the answer is essentially yes, in the sense that without this property, the resulting $(d-1)$-dimensional integrand is generally not smooth, having square-root or other singularities. The square-root singularity is generically enough to prevent the preintegrated function from belonging to the mixed derivative spaces typically used in Quasi-Monte Carlo or Sparse Grid analysis.

Keywords Preintegration · Smoothing · Conditional sampling · Numerical integration

A. D. Gilbert (✉) · F. Y. Kuo · I. H. Sloan
School of Mathematics and Statistics, UNSW Sydney, Sydney NSW, 2052, Australia
e-mail: alexander.gilbert@unsw.edu.au

F. Y. Kuo
e-mail: f.kuo@unsw.edu.au

I. H. Sloan
e-mail: i.sloan@unsw.edu.au

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_9

1 Introduction

Preintegration is a method for numerical integration over $\mathbb{R}^d$, where d may be large, in the presence of "kinks" (i.e., discontinuities in the gradients) or "jumps" (i.e., discontinuities in the function values). In this method, one of the variables is integrated out in a "preintegration" step, with the aim of creating a smooth integrand over $\mathbb{R}^{d-1}$. Smoothness is important if the intention is to approximate the $(d-1)$-dimensional integral by a method that relies on some smoothness of the integrand, such as the Quasi-Monte Carlo (QMC) method [6] or Sparse Grid (SG) method [5].

Integrands with kinks and jumps arise in option pricing, because an option is normally considered worthless if the value falls below a predetermined strike price. In the case of a continuous payoff function this introduces a kink, while in the case of a binary or other digital option it introduces a jump. Integrands with jumps also arise in computations of cumulative probability distributions, see [7].

In this paper we consider the version of preintegration for functions with kinks or jumps presented in the recent papers [10–12], in which the emphasis was on a rigorous proof of smoothness of the preintegrated $(d-1)$-dimensional integrand under appropriate conditions, where smoothness is determined by membership of a certain mixed derivative Sobolev space.

A key assumption in [10–12] was that the smooth function (the function ϕ in (2) below) underlying the kink or jump is *strictly monotone* with respect to the special variable chosen for the preintegration step, when all other variables are held fixed. While a satisfactory analysis was obtained under that assumption, it was not clear from the analysis in [10–12] whether or not the monotonicity assumption is in some sense necessary. That is the question we address in the present paper. The short answer is that *the monotonicity condition is necessary, in that in the absence of monotonicity, the integrand typically has square-root or other singularities*. Although a square-root singularity is better behaved than a jump discontinuity, we shall see in Sect. 2 (see Example 5) that the square-root singularity (defined precisely in Definition 1) is generically enough to prevent the preintegrated function from belonging to the mixed derivative spaces typically used in QMC or SG analysis.

1.1 Related Work

A similar method has already appeared as a practical tool in many other papers, often under the heading "conditional sampling", see [9], Lemma 7.2 and preceding comments in [1], and recent papers [15, 16] by L'Ecuyer and colleagues. Also relevant are root-finding strategies for identifying where the payoff is positive, see a remark in [2, 13, 18]. For other "smoothing" methods, see [3, 19].

The goal in conditional sampling is usually to decrease the variance of the integrand, motivated by the idea that if the Monte Carlo method is the chosen method for evaluating the integral then reducing the variance will certainly reduce the root

mean square expected error. The reality of variance reduction in the preintegration context was explored analytically in Sect. 4 of [12]. But if cubature methods are used that depend on smoothness of the integrand, as with QMC and SG methods, then variance reduction is not the only consideration. In the present work the focus is on smoothness of the resulting integrand.

1.2 The Problem

For the rest of the paper we will follow the setting of [12]. The problem addressed in [12] was the approximate evaluation of the d-dimensional integral

$$I_d f \;:=\; \int_{\mathbb{R}^d} f(\boldsymbol{y})\rho_d(\boldsymbol{y})\,\mathrm{d}\boldsymbol{y} \;=\; \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} f(y_1,\ldots,y_d)\,\rho_d(\boldsymbol{y})\,\mathrm{d}y_1\cdots\mathrm{d}y_d, \quad (1)$$

with

$$\rho_d(\boldsymbol{y}) \;:=\; \prod_{k=1}^{d}\rho(y_k),$$

where ρ is a continuous and strictly positive probability density function on $\mathbb{R}$, and f is a real-valued function of the form

$$f(\boldsymbol{y}) \;=\; \theta(\boldsymbol{y})\,\mathrm{ind}\big(\phi(\boldsymbol{y})\big), \quad (2)$$

or more generally

$$f(\boldsymbol{y}) \;=\; \theta(\boldsymbol{y})\,\mathrm{ind}\big(\phi(\boldsymbol{y})-t\big), \quad (3)$$

where θ and ϕ are smooth functions in some sense, $\mathrm{ind}(\cdot)$ is the indicator function which has the value 1 if the argument is positive and 0 otherwise, and t is an arbitrary real number. When $t = 0$ and $\theta = \phi$ we have $f(\boldsymbol{y}) = \max(\phi(\boldsymbol{y}), 0)$ and thus we have the familiar kink seen in option pricing through the occurrence of a strike price. When θ and ϕ are different (for example, when $\theta(\boldsymbol{y}) = 1$) we have a structure that includes digital options.

The key assumption on the smooth function ϕ in [12] was that it has a positive partial derivative with respect to some well chosen variable y_j (and so is an increasing function of y_j); that is, we assume that for one special choice of $j \in \{1,\ldots,d\}$ we have

$$\frac{\partial\phi}{\partial y_j}(\boldsymbol{y}) > 0 \qquad \text{for all} \quad \boldsymbol{y}\in\mathbb{R}^d. \quad (4)$$

In other words, ϕ is monotone increasing with respect to y_j when all variables other than y_j are held fixed.

With the variable y_j chosen to satisfy this condition, the preintegration step is to evaluate

$$(P_j f)(\boldsymbol{y}_{-j}) := \int_{-\infty}^{\infty} f(y_j, \boldsymbol{y}_{-j})\, \rho(y_j)\, \mathrm{d}y_j, \tag{5}$$

where $\boldsymbol{y}_{-j} \in \mathbb{R}^{d-1}$ denotes all the components of $\boldsymbol{y}$ other than y_j. Once $(P_j f)(\boldsymbol{y}_{-j})$ is known we can evaluate $I_d f$ as the $(d-1)$-dimensional integral

$$I_d f = \int_{\mathbb{R}^{d-1}} (P_j f)(\boldsymbol{y}_{-j})\, \rho_{d-1}(\boldsymbol{y}_{-j})\, \mathrm{d}\boldsymbol{y}_{-j}, \tag{6}$$

which can be done efficiently if $(P_j f)(\boldsymbol{y}_{-j})$ is smooth. In the implementation of preintegration, note that if the integral (6) is to be evaluated by an N-point cubature rule, then the preintegration step in (5) needs to be carried out for N different values of $\boldsymbol{y}_{-j}$.

The key is the preintegration step. Because of the monotonicity assumption (4), for each $\boldsymbol{y}_{-j} \in \mathbb{R}^{d-1}$, there is at most one value of the integration variable y_j such that $\phi(y_j, \boldsymbol{y}_{-j}) = t$, where we recall that t is the parameter in the general form of $f(\boldsymbol{y})$ as in (3). We denote that value of y_j, if it exists, by $\xi(\boldsymbol{y}_{-j}) = \xi_t(\boldsymbol{y}_{-j})$ so that $\phi(\xi(\boldsymbol{y}_{-j}), \boldsymbol{y}_{-j}) = t$. Under the condition (4), it follows from the implicit function theorem that $\xi(\boldsymbol{y}_{-j})$ is smooth if ϕ is smooth. Then we can write the preintegration step as

$$(P_j f)(\boldsymbol{y}_{-j}) = \int_{\xi(\boldsymbol{y}_{-j})}^{\infty} \theta\big(y_j, \boldsymbol{y}_{-j}\big)\, \rho(y_j)\, \mathrm{d}y_j, \tag{7}$$

which is a smooth function of $\boldsymbol{y}_{-j}$ if θ is smooth.

If ϕ is strictly decreasing instead of increasing with respect to y_j, then analogous arguments show that $P_j f$ is again smooth if θ is smooth. In this case, the limits of the integral in (7) are $-\infty$ to $\xi(\boldsymbol{y}_{-j})$. Clearly, the essential property is that ϕ is monotonic with respect to y_j, which allows the implicit function theorem to be used.

For the remainder of this paper, we consider what happens after preintegration if monotonicity fails, i.e., if $(\partial\phi/\partial y_j)(\boldsymbol{y}^*) = 0$ at some point $\boldsymbol{y}^* \in \mathbb{R}^d$.

1.3 Informative Examples

We now illustrate the success and failure of the preintegration process with some simple examples. In these examples, we take $d = 2$ and $t = 0$, and choose ρ to be the standard normal probability density, $\rho(y) = \exp(-y^2/2)/\sqrt{2\pi}$. We also initially take $\theta(y_1, y_2) = 1$, and comment on other choices at the end of the section.

To help in our discussion of the examples, we define the following subsets related to a function $\phi : \mathbb{R}^d \to \mathbb{R}$. The *zero level set* of ϕ is $\{\boldsymbol{y} \in \mathbb{R}^d : \phi(\boldsymbol{y}) = 0\}$ and the *positivity set* of ϕ is $\{\boldsymbol{y} \in \mathbb{R}^d : \phi(\boldsymbol{y}) > 0\}$.

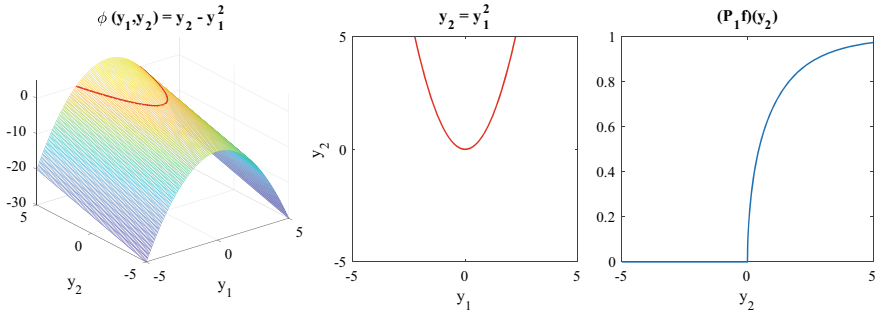


Fig. 1 Illustrations for Example 1

Example 1 In this example, we take

$$\phi(y_1, y_2) \;=\; y_2 - y_1^2,$$

see Fig. 1 (left). The zero level set of this function is the parabolic curve $y_2 = y_1^2$, see Fig. 1 (middle). The positivity set of ϕ is the open region above the parabola.

If we take the special variable to be y_2 (i.e., if we take $j = 2$) then the monotonicity condition (4) is satisfied, and the preintegration step is truly smoothing. Specifically, we see that

$$(P_2 f)(y_1) \;=\; \int_{y_1^2}^{\infty} \rho(y_2)\,\mathrm{d}y_2 \;=\; 1 - \Phi(y_1^2),$$

where $\Phi(x) := \int_{-\infty}^{x} \rho(y)\,\mathrm{d}y$ is the standard normal cumulative distribution. Thus $(P_2 f)(y_1)$ is a smooth function for all $y_1 \in \mathbb{R}$, and $I_2 f$ is the integral of a smooth integrand over the real line,

$$I_2 f \;=\; \int_{-\infty}^{\infty} (P_2 f)(y_1)\,\rho(y_1)\,\mathrm{d}y_1 \;=\; \int_{-\infty}^{\infty} \big(1 - \Phi(y_1^2)\big)\,\rho(y_1)\,\mathrm{d}y_1.$$

If on the other hand we take the special variable to be y_1 (i.e., take $j = 1$) so that the monotonicity condition (4) is violated, then we have

$$(P_1 f)(y_2) \;=\; \begin{cases} 0 & \text{if } y_2 \le 0, \\ \displaystyle\int_{-\sqrt{y_2}}^{\sqrt{y_2}} \rho(y_1)\,\mathrm{d}y_1 \;=\; \Phi(\sqrt{y_2}) - \Phi(-\sqrt{y_2}) & \text{if } y_2 > 0. \end{cases}$$

The graph of $(P_1 f)(y_2)$, shown in Fig. 1 (right), reveals that there is a singularity at $y_2 = 0$. To see the nature of the singularity, note that since $\rho(y_1) = \rho(0)\exp(-y_1^2/2) = \rho(0) + \mathcal{O}(y_1^2)$ as $y_1 \to 0$, we can write

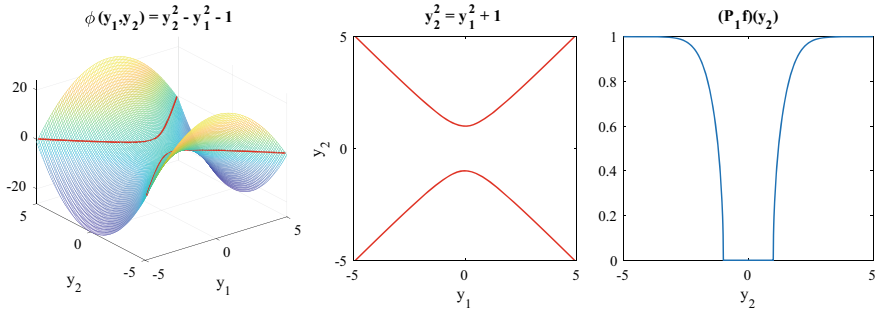


Fig. 2 Illustrations for Example 2

$$(P_1 f)(y_2) \;=\; \begin{cases} 0 & \text{if } y_2 \le 0, \\ \displaystyle\int_{-\sqrt{y_2}}^{\sqrt{y_2}} \rho(y_1)\,\mathrm{d}y_1 \;=\; 2\sqrt{y_2}\,\rho(0) + \mathcal{O}\big(y_2^{3/2}\big) & \text{if } y_2 > 0. \end{cases} \tag{8}$$

Thus in this simple example, $(P_1 f)(y_2)$ is not a smooth function of y_2, having a square-root singularity, and hence an infinite one-sided derivative as $y_2 \to 0^+$.

Example 2 In this example, we take

$$\phi(y_1, y_2) \;=\; y_2^2 - y_1^2 - 1,$$

see Fig. 2 (left). The zero level set of ϕ is now the hyperbola $y_2^2 = y_1^2 + 1$, see Fig. 2 (middle), and the positivity set is the union of the open regions above and below the upper and lower branches, respectively. Taking $j = 1$, we see that monotonicity again fails, and that specifically,

$$(P_1 f)(y_2)$$
$$= \begin{cases} 0 & \text{if } y_2 \in [-1, 1], \\ \displaystyle\int_{-\sqrt{y_2^2-1}}^{\sqrt{y_2^2-1}} \rho(y_1)\,\mathrm{d}y_1 \;=\; 2\sqrt{y_2^2 - 1}\,\rho(0) + \mathcal{O}\big((y_2^2 - 1)^{3/2}\big) & \text{if } |y_2| > 1. \end{cases}$$

Its graph is shown in Fig. 2 (right). Again we see square-root singularities, this time two of them.

Example 3 Here we take

$$\phi(y_1, y_2) \;=\; y_2^2 - y_1^2,$$

see Fig. 3 (left). The zero level set is now the pair of lines $y_2 = \pm y_1$, see Fig. 3 (middle), and the positivity set is the open region above and below the crossed lines. This time $P_1 f$ is given by

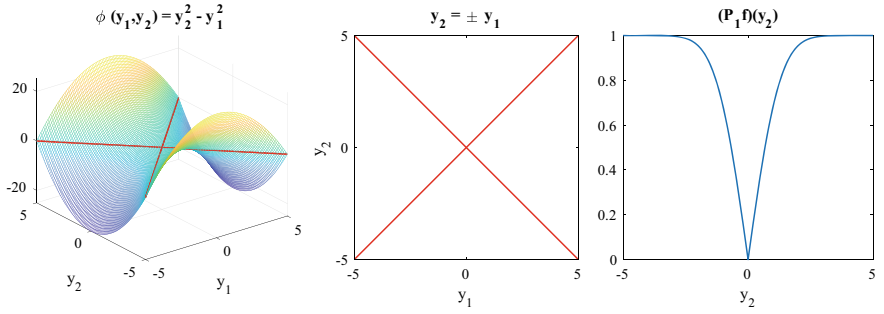


Fig. 3 Illustrations for Example 3

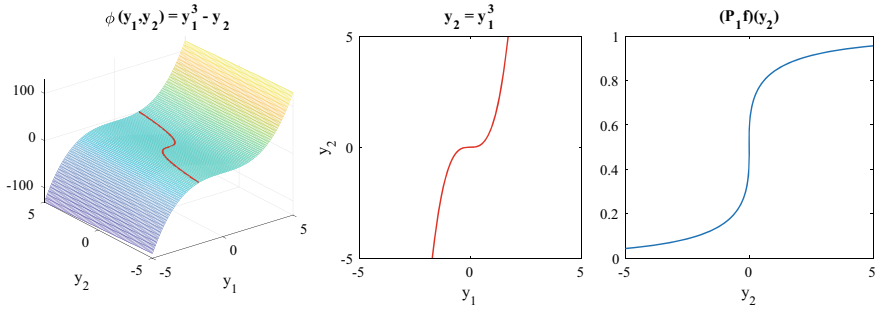


Fig. 4 Illustrations for Example 4

$$(P_1 f)(y_2) \;=\; \int_{-|y_2|}^{|y_2|} \rho(y_1)\,\mathrm{d}y_1 \;=\; 2|y_2|\,\rho(0) + \mathcal{O}\big(|y_2|^3\big),$$

revealing in Fig. 3 (right) a different kind of singularity (a jump discontinuity in the first derivative), but one still unfavorable for numerical integration.

Example 3 is rather special, in that the preintegration is performed on a line that touches a saddle at its critical point (the "flat point" of the saddle). Example 4 below illustrates another situation, one that is in some ways similar to Example 1, but one perhaps less likely to be seen in practice.

Example 4 Here we consider

$$\phi(y_1, y_2) \;=\; y_1^3 - y_2,$$

see Fig. 4 (left). The zero level set of ϕ is the graph of $y_2 = y_1^3$, see Fig. 4 (middle), and the positivity set is the unbounded domain to the right of the curve. We see that

$$(P_1 f)(y_2) \;=\; \int_{-\infty}^{y_2^{1/3}} \rho(y_1)\,\mathrm{d}y_1 \;=\; \int_{-\infty}^{0} \rho(y_1)\,\mathrm{d}y_1 + \int_{0}^{y_2^{1/3}} \rho(y_1)\,\mathrm{d}y_1$$
$$= \frac{1}{2} + y_2^{1/3}\rho(0) + \mathcal{O}\big(|y_2|\big),$$

which holds regardless of the sign of y_2. The graph of $P_1 f$ in Fig. 4 (right) displays the cube-root singularity at $y_2 = 0$.

In each of the above examples we took $\theta(y_1, y_2) = 1$. Other choices for θ are generally not more interesting. An exception is the choice $\theta(y_1, y_2) = \phi(y_1, y_2)$, which yields a kink rather than a jump because

$$\phi(\boldsymbol{y})\,\mathrm{ind}(\phi(\boldsymbol{y})) \;=\; \max(\phi(\boldsymbol{y}), 0),$$

and so leads to a weaker singularity. For example, for $f(y_1, y_2) = \max(\phi(y_1, y_2), 0)$ with ϕ as in Example 1, we obtain instead of (8)

$$(P_1 f)(y_2) \;=\; \begin{cases} 0 & \text{if } y_2 \le 0, \\ \displaystyle\int_{-\sqrt{y_2}}^{\sqrt{y_2}} (y_2 - y_1^2)\,\rho(y_1)\,\mathrm{d}y_1 \;=\; \frac{4}{3}\,y_2^{3/2}\,\rho(0) + \mathcal{O}\big(y_2^{5/2}\big) & \text{if } y_2 > 0. \end{cases}$$

With the recognition that kinks lead to less severe singularities than jumps, but located at the same places, from now on we shall for simplicity consider only the case $\theta(\boldsymbol{y}) = 1$.

1.4 Outline of This Paper

In Sect. 2 we study theoretically the smoothness of the preintegrated function, assuming that the original d-variate function is $f(\boldsymbol{y}) = \mathrm{ind}(\phi(\boldsymbol{y}) - t)$, with ϕ smooth but not monotone. We prove that the behavior seen in the above informative examples is typical, and give a precise definition of a square-root singularity in Definition 1. We also show in Example 5 that for $d \ge 3$ a square-root singularity is generically enough to preclude the preintegrated function from belonging to any of the typical mixed derivative spaces used to study convergence of QMC or SG methods.

Section 3 contains a numerical experiment for a high-dimensional integrand that allows both monotone and non-monotone choices for the preintegration variable. Section 4 gives brief conclusions.

2 Smoothness Theorems in d Dimensions

In the general d-dimensional setting, we take $\theta \equiv 1$ and use the general form (3) with arbitrary $t \in \mathbb{R}$. Thus now we consider

$$f(\boldsymbol{y}) := f_t(\boldsymbol{y}) := \operatorname{ind}\big(\phi(\boldsymbol{y}) - t\big), \quad \boldsymbol{y} \in \mathbb{R}^d. \tag{9}$$

A natural setting in which t can take any value is in the computation of the (complementary) cumulative distribution function of a random variable of the form $X = \phi(\boldsymbol{Y})$, as in [7], where $\boldsymbol{Y} = (Y_1, Y_2, \ldots, Y_d) \in \mathbb{R}^d$ is a vector of independent real-valued random variables, with realizations $\boldsymbol{y} = (y_1, y_2, \ldots, y_d)$. In the case of option pricing, varying t corresponds to varying the strike price.

For simplicity, in this section we shall always take the special preintegration variable to be y_1, so we fix $j = 1$. The task, assuming that ϕ in (9) has smoothness at least $C^2(\mathbb{R}^d)$, is to study the smoothness of

$$(P_1 f_t)(\boldsymbol{y}_{-1}) := \int_{-\infty}^{\infty} f_t(y_1, \boldsymbol{y}_{-1})\, \rho(y_1)\, \mathrm{d}y_1 = \int_{-\infty}^{\infty} \operatorname{ind}\big(\phi(\boldsymbol{y}) - t\big)\, \rho(y_1)\, \mathrm{d}y_1, \tag{10}$$

where $\rho \in C(\mathbb{R})$ is a general probability density function with $\operatorname{supp}(\rho) = \mathbb{R}$.

To gain a first insight into the role of the parameter t in (9), it is useful to observe that for the examples in Sect. 1.3 a variation in t can change the position and even the nature of the singularity in $P_1 f_t$, but does not necessarily eliminate the singularity. For a general $t \in \mathbb{R}$ and ϕ as in Example 1, we easily find that (8) is replaced by

$$(P_1 f_t)(y_2) = \begin{cases} 0 & \text{if } y_2 \le t, \\ \displaystyle\int_{-\sqrt{y_2 - t}}^{\sqrt{y_2 - t}} \rho(y_1)\, \mathrm{d}y_1 & \text{if } y_2 > t, \end{cases}$$

so that the graph of $P_1 f_t$ is simply translated with the singularity now occurring at $y_2 = t$ instead of $y_2 = 0$. The situation is the same for ϕ as in Example 4.

For ϕ as in Example 2, the choice $t = -1$ recovers Example 3, while for $t > -1$ we find

$$(P_1 f_t)(y_2) = \begin{cases} 0 & \text{if } y_2 \in [-\sqrt{1+t}, \sqrt{1+t}\,], \\ \displaystyle\int_{-\sqrt{y_2^2 - 1 - t}}^{\sqrt{y_2^2 - 1 - t}} \rho(y_1)\, \mathrm{d}y_1 & \text{if } |y_2| > \sqrt{1+t}\,, \end{cases}$$

thus in this case $(P_1 f_t)(y_2)$ has square-root singularities at $y_2 = \pm\sqrt{1+t}$. For the case $t < -1$ (which we leave to the reader) $P_1 f_t$ has no singularity.

In [12] it was proved that $P_1 f_t$ has the same smoothness as ϕ, provided that

$$\frac{\partial \phi}{\partial y_1}(\boldsymbol{y}) > 0 \qquad \text{for all} \quad \boldsymbol{y} \in \mathbb{R}^d, \tag{11}$$

together with some other technical conditions, see [12, Theorems 2 and 3]. Here the important property is that ϕ is monotone, since if ϕ is instead monotone *decreasing*, i.e., $(\partial\phi/\partial y_1)(\boldsymbol{y}) < 0$ for all $\boldsymbol{y} \in \mathbb{R}^d$, then simple modifications to the arguments and technical conditions in [12] show again that $P_1 f_t$ has the same smoothness as ϕ.

Here we are interested in the situation when ϕ is **not** monotone with respect to y_1 for all $\boldsymbol{y}_{-1}$. In that case there is at least one point, say $\boldsymbol{y}^* = (y_1^*, \boldsymbol{y}_{-1}^*) \in \mathbb{R}^d$, at which $(\partial\phi/\partial y_1)(\boldsymbol{y}^*) = 0$. At such a point, the gradient of ϕ is either zero or orthogonal to the y_1 axis. If t in (9) has the value $t = \phi(\boldsymbol{y}^*)$, then there is generically a singularity of some kind in $P_1 f_t$ at the point $\boldsymbol{y}_{-1}^* \in \mathbb{R}^{d-1}$. If $t \neq \phi(\boldsymbol{y}^*)$, then there is in general no singularity in $P_1 f_t$ at the point $\boldsymbol{y}_{-1}^* \in \mathbb{R}^{d-1}$, but if t is allowed to vary, then the risk of encountering a near-singularity is high.

Theorem 1 below states a general result for the existence and the nature of the singularities induced in $P_1 f_t$ in the common situation in which the second derivative of ϕ with respect to y_1 is non-zero at $\boldsymbol{y} = \boldsymbol{y}^*$, the point at which the first derivative with respect to y_1 is zero. If the second derivative is not zero but a higher partial derivative with respect to y_1 is non-zero, then by a similar argument a weaker singularity arises. For simplicity, we shall concentrate on the case in which the second partial derivative is non-zero.

First, we make a formal definition of "square-root singularity", as encountered in Examples 1 and 2. Note that in higher dimensions, an isolated singularity can be approached from multiple directions.

Definition 1 A function $g \in C(\mathbb{R}^{d-1})$, with $d \geq 2$, is defined to have a *square root singularity* at $\boldsymbol{y}^* \in \mathbb{R}^{d-1}$ in the direction $\boldsymbol{z} \in \mathbb{R}^{d-1}$, $\|\boldsymbol{z}\| = 1$, if

$$\lim_{\tau \to 0^+} \frac{g(\boldsymbol{y}^* + \tau \boldsymbol{z}) - g(\boldsymbol{y}^*)}{\sqrt{\tau}} = \alpha, \quad \text{for some } 0 \neq \alpha \in \mathbb{R}.$$

Theorem 1 *Let $\phi \in C^2(\mathbb{R}^d)$, $\rho \in C(\mathbb{R})$ with $\operatorname{supp}(\rho) = \mathbb{R}$, and assume that $\boldsymbol{y}^* = (y_1^*, \boldsymbol{y}_{-1}^*) \in \mathbb{R}^d$ is such that*

$$\frac{\partial \phi}{\partial y_1}(\boldsymbol{y}^*) = 0, \quad \frac{\partial^2 \phi}{\partial y_1^2}(\boldsymbol{y}^*) \neq 0, \quad \textit{and} \quad \nabla\phi(\boldsymbol{y}^*) \neq \boldsymbol{0}. \tag{12}$$

Define $t := \phi(\boldsymbol{y}^*)$. Then the function $(P_1 f_t)(\boldsymbol{y}_{-1})$ as defined in (10) has a square-root singularity at $\boldsymbol{y}_{-1}^* \in \mathbb{R}^{d-1}$ in any direction in $\mathbb{R}^{d-1}$ that has a positive inner product with $\nabla_{-1}\phi(\boldsymbol{y}^*) := ((\partial\phi/\partial y_2)(\boldsymbol{y}^*), (\partial\phi/\partial y_3)(\boldsymbol{y}^*), \ldots, (\partial\phi/\partial y_d)(\boldsymbol{y}^*))$.

Proof Since $\nabla\phi(\boldsymbol{y}^*)$ is not zero, and has no component in the direction of the y_1 axis, it follows that $\nabla\phi(\boldsymbol{y}^*)$ can be written as $(0, \nabla_{-1}\phi(\boldsymbol{y}^*))$, where $\nabla_{-1}\phi(\boldsymbol{y}^*) = \nabla_{-1}\phi(y_1^*, \boldsymbol{y}_{-1}^*)$ is a non-zero vector in $\mathbb{R}^{d-1}$ orthogonal to the y_1 axis. Note that as

$\boldsymbol{y}_{-1}$ changes in a neighborhood of $\boldsymbol{y}^*_{-1}$, with y_1 held fixed, the function $\phi(y_1^*, \boldsymbol{y}_{-1})$ is increasing in the direction $\nabla_{-1}\phi(\boldsymbol{y}^*)$, and also in the direction of an arbitrary unit vector $\boldsymbol{z}$ in $\mathbb{R}^{d-1}$ that has a positive inner product with $\nabla_{-1}\phi(\boldsymbol{y}^*)$. Our aim now is to explore the nature of $P_1 f_t$ on the line through $\boldsymbol{y}^*_{-1} \in \mathbb{R}^{d-1}$ in the direction of $\boldsymbol{z}$.

For simplicity of presentation, and without loss of generality, we assume from now on that the unit vector $\boldsymbol{z}$ points in the direction of the positive y_2 axis. This allows us to write $\boldsymbol{y} = (y_1, y_2, y_3^*, \ldots, y_d^*) =: (y_1, y_2)$, temporarily holding fixed and ignoring all components other than the first two. In this 2-dimensional setting we know that

$$\frac{\partial \phi}{\partial y_1}(y_1^*, y_2^*) = 0, \quad \frac{\partial \phi}{\partial y_2}(y_1^*, y_2^*) > 0, \quad \text{and} \quad \frac{\partial^2 \phi}{\partial y_1^2}(y_1^*, y_2^*) \neq 0. \tag{13}$$

Since $(\partial\phi/\partial y_2)(y_1, y_2)$ is continuous and positive at $(y_1, y_2) = (y_1^*, y_2^*)$, it follows, for sufficiently small $\delta > 0$, that for each $y_1 \in [y_1^* - \delta, y_1^* + \delta]$ there is at most one value of y_2 such that $\phi(y_1, y_2) = t$. For that unique value we write $y_2 = \zeta(y_1)$, hence by construction we have $\phi(y_1, \zeta(y_1)) = t$, and $\zeta(y_1^*) = y_2^*$.

(Note that ζ here is the implicit function for a two-dimensional problem, and is a univariate function of the preintegration variable y_1, whereas ξ in (7) is the implicit function from the preintegration process, giving y_1 as a function of the remaining variables $y_2, y_3, \ldots, y_d$.)

From the implicit function theorem (or by implicit differentiation of $\phi(y_1, \zeta(y_1)) = t$ with respect to y_1) we obtain

$$\zeta'(y_1) = -\frac{(\partial\phi/\partial y_1)(y_1, \zeta(y_1))}{(\partial\phi/\partial y_2)(y_1, \zeta(y_1))}, \tag{14}$$

in which the denominator is positive in a neighborhood of y_1^*. From this and the first condition in (13), it follows that

$$\zeta'(y_1^*) = -\frac{(\partial\phi/\partial y_1)(y_1^*, y_2^*)}{(\partial\phi/\partial y_2)(y_1^*, y_2^*)} = 0.$$

Differentiating (14) using the product rule and the chain rule and then setting $y_1 = y_1^*$ (so that several terms vanish), we obtain using the second condition in (13) that

$$\zeta''(y_1^*) = -\frac{(\partial^2\phi/\partial y_1^2)(y_1^*, y_2^*)}{(\partial\phi/\partial y_2)(y_1^*, y_2^*)} \neq 0.$$

We now assume that $(\partial^2\phi/\partial y_1^2)(y_1^*, y_2^*) < 0$, from which it follows that $\zeta''(y_1^*)$ is positive; the case $(\partial^2\phi/\partial y_1^2)(y_1^*, y_2^*) > 0$ is similar. Taylor's theorem with remainder gives

$$\begin{aligned}\zeta(y_1) &= \zeta(y_1^*) + \tfrac{1}{2}(y_1 - y_1^*)^2\,\zeta''(y_1^*)\,(1 + o(1)) \\ &= y_2^* + \tfrac{1}{2}(y_1 - y_1^*)^2\,\zeta''(y_1^*)\,(1 + o(1)),\end{aligned} \tag{15}$$

where $o(1) \to 0$ as $|y_1 - y_1^*| \to 0$. Thus $\zeta(y_1)$ is a convex function in a neighborhood of y_1^*.

Given y_2 in a neighborhood of y_2^*, our task now is to evaluate the contribution to the integral

$$(P_1 f_t)(y_2) = \int_{-\infty}^{\infty} \mathrm{ind}\big(\phi(y_1, y_2) - t\big)\rho(y_1)\,\mathrm{d}y_1$$

from a neighborhood of y_1^*. Thus we need to find the set of y_1 values in a neighborhood of y_1^* for which $\phi(y_1, y_2) > t$. Because of (15), the set is either empty, or is the open interval with extreme points given by the solutions y_1 of $\zeta(y_1) = y_2$, i.e.,

$$y_2^* + \tfrac{1}{2}(y_1 - y_1^*)^2\,\zeta''(y_1^*)\,(1 + o(1)) = y_2,$$

implying

$$(y_1 - y_1^*)^2 = \frac{2(y_2 - y_2^*)}{\zeta''(y_1^*)\,(1 + o(1))} = \frac{2(y_2 - y_2^*)}{\zeta''(y_1^*)}(1 + o(1)).$$

There is no solution for $y_2 < y_2^*$, while for $y_2 > y_2^*$ the solutions are

$$y_1 = y_1^* \pm c\sqrt{y_2 - y_2^*}\,(1 + o(1)),$$

with $c := \sqrt{2/\zeta''(y_1^*)}$. Thus the contribution to $P_1 f_t(y_2)$ from the neighborhood of y_2^* is zero for $y_2 < y_2^*$, and for $y_2 > y_2^*$ is

$$\begin{aligned}&\int_{y_1^* - c\sqrt{y_2 - y_2^*}\,(1 + o(1))}^{y_1^* + c\sqrt{y_2 - y_2^*}\,(1 + o(1))} \rho(y_1)\,\mathrm{d}y_1 \\ &\quad = \Phi\Big(y_1^* + c\sqrt{y_2 - y_2^*}\,(1 + o(1))\Big) - \Phi\Big(y_1^* - c\sqrt{y_2 - y_2^*}\,(1 + o(1))\Big) \\ &\quad = c\sqrt{y_2 - y_2^*}\,(1 + o(1))(\rho(a_+) + \rho(a_-)) = O\Big(\sqrt{y_2 - y_2^*}\Big),\end{aligned}$$

where $o(1) \to 0$ and $a_\pm \to y_1^*$ as $y_2 - y_2^* \to 0^+$. Here we have used the fact that $\Phi \in C^1(\mathbb{R})$, since $\Phi' = \rho \in C(\mathbb{R})$. This allowed us to expand Φ around y_1^* using the mean value theorem as follows:

$$\Phi\Big(y_1^* + c\sqrt{y_2 - y_2^*}\,(1 + o(1))\Big) = \Phi(y_1^*) + c\sqrt{y_2 - y_2^*}\,(1 + o(1))\rho(a_+)$$

for some $a_+ \in (y_1^*, y_1^* + c\sqrt{y_2 - y_2^*}\,(1 + o(1)))$, and similarly for $\Phi(y_1^* - c\sqrt{y_2 - y_2^*}\,(1 + o(1)))$ with $a_- \in (y_1^* - c\sqrt{y_2 - y_2^*}\,(1 + o(1)), y_1^*)$. We conclude

that there is a square-root singularity in $P_1 f_t$ in the direction of the positive y_2 axis, which is the direction of the unit vector $\boldsymbol{z}$, which by assumption has a positive inner product with $\nabla_{-1}\phi(\boldsymbol{y}^*)$. □

Remark 1 Theorem 1 gives specific conditions for the existence of a square-root singularity after preintegration. However, under slightly different conditions it can be shown, following a similar proof technique, that singularities of a different nature may also occur. The key driver of the nature of the singularity in Theorem 1 is the fact that $(\partial\phi/\partial y_1)(\boldsymbol{y}^*) = 0$ and $(\partial^2\phi/\partial y_1^2)(\boldsymbol{y}^*) \neq 0$. If instead $(\partial\phi/\partial y_1)(\boldsymbol{y}^*) = (\partial^2\phi/\partial y_1^2)(\boldsymbol{y}^*) = 0$ but $(\partial^3\phi/\partial y_1^3)(\boldsymbol{y}^*) \neq 0$, then (15) becomes $\zeta(y_1) = y_2^* + \frac{1}{6}(y_1 - y_1^*)^3\zeta'''(y_1^*)(1 + o(1))$. In this case, it is easy to see that there will be a *cube-root* singularity at $\boldsymbol{y}^*$, with behavior similar to Example 4.

Remark 2 The assumption $\phi \in C^2(\mathbb{R}^d)$ can be weakened to ϕ continuously differentiable and $\partial^2\phi/\partial y_1^2$ continuous in a neighborhood of $\boldsymbol{y}^*$. On the other hand, it is clear that *strengthening* the assumptions on the differentiability of ϕ and ρ will neither remove the singularity nor change its nature.

Remark 3 We now return to consider the examples in Sect. 1.3 in the context of Theorem 1.

- For ϕ as in Example 1, we have $(\partial\phi/\partial y_1)(y_1, y_2) = -2y_1$, $(\partial^2\phi/\partial y_1^2)(y_1, y_2) = -2 \neq 0$, and $\nabla\phi(y_1, y_2) = (-2y_1, 1) \neq (0, 0)$. Thus (12) holds, e.g., with $\boldsymbol{y}^* = (0, 0)$ and $t = \phi(\boldsymbol{y}^*) = 0$, and $P_1 f_t$ indeed displays the predicted square-root singularity at $y_2 = 0$, see Fig. 1.
- For ϕ as in Example 2, we have $(\partial\phi/\partial y_1)(y_1, y_2) = -2y_1$, $(\partial^2\phi/\partial y_1^2)(y_1, y_2) = -2 \neq 0$, and $\nabla\phi(y_1, y_2) = (-2y_1, 2y_2)$. Thus (12) holds, e.g., with $\boldsymbol{y}^* = (0, \pm 1)$ and $t = \phi(\boldsymbol{y}^*) = 0$, and $P_1 f_t$ indeed shows the predicted square-root singularities at $y_2 = \pm 1$, see Fig. 2.
- For ϕ as in Example 3, we have the same derivative expressions as in Example 2. Thus (12) holds, e.g., again with $\boldsymbol{y}^* = (0, \pm 1)$, but now $t = \phi(\boldsymbol{y}^*) = 1$, and we effectively recover Example 2 with square-root singularities for $P_1 f_t$ at $y_2 = \pm 1$. However, if we consider instead the point $\boldsymbol{y}^\dagger = (0, 0)$ and $t = \phi(\boldsymbol{y}^\dagger) = 0$, as in Fig. 3, then we have $\nabla\phi(\boldsymbol{y}^\dagger) = 0$ so the non-vanishing gradient condition in (12) fails and Theorem 1 does not apply at this point $\boldsymbol{y}^\dagger$. In this case we actually observe an absolute-value singularity for $P_1 f_t$ at $y_2 = 0$ rather than a square-root singularity.
- For ϕ as in Example 4, we have $(\partial\phi/\partial y_1)(y_1, y_2) = 3y_1^2$, $(\partial^2\phi/\partial y_1^2)(y_1, y_2) = 6y_1$, and $\nabla\phi(y_1, y_2) = (3y_1^2, -1) \neq (0, 0)$. It is impossible to satisfy both the first and second conditions in (12) so Theorem 1 does not apply anywhere. In particular, at the point $\boldsymbol{y}^\dagger = (0, 0)$ and $t = \phi(\boldsymbol{y}^\dagger) = 0$, as in Fig. 4, we have $(\partial^2\phi/\partial y_1^2)(\boldsymbol{y}^\dagger) = 0$, and in consequence (given that the third derivative does not vanish) $P_1 f_t$ has a cube-root singularity at $y_2 = 0$ rather than a square-root singularity.

The following example with $d = 3$ demonstrates concretely that, for $d \geq 3$ (and hence $d - 1 \geq 2$), a square-root singularity is generically enough to prevent the

preintegrated function from belonging to any of the mixed derivative spaces conventionally assumed in QMC and SG convergence analysis.

Example 5 Consider

$$\phi(y_1, y_2, y_3) \;=\; 1 - y_1^2 - (y_2 - y_3 - 1)^2,$$

so $(\partial\phi/\partial y_1)(y_1, y_2, y_3) = -2y_1, (\partial^2\phi/\partial y_1^2)(y_1, y_2, y_3) = -2$, and $\nabla\phi(y_1, y_2, y_3) = (-2y_1, -2(y_2 - y_3 - 1), 2(y_2 - y_3 - 1))$. Taking $\boldsymbol{y}^* = (0, 0, 0)$ yields

$$t = \phi(\boldsymbol{y}^*) = 0, \quad \frac{\partial\phi}{\partial y_1}(\boldsymbol{y}^*) = 0, \quad \frac{\partial^2\phi}{\partial y_1^2}(\boldsymbol{y}^*) = -2 \neq 0, \quad \nabla\phi(\boldsymbol{y}^*) = (0, 2, -2) \neq \mathbf{0}.$$

Thus all three conditions of (12) are satisfied, and Theorem 1 tells us that $(P_1 f_t)(y_2, y_3)$ as defined in (10) has a square-root singularity at $(y_2, y_3) = (0, 0)$ in any direction in $\mathbb{R}^2$ that has a positive inner product with $(2, -2)$, i.e., for any direction in the half plane $y_2 > y_3$.

We now find $P_1 f_t$ explicitly, by carrying out the preintegration of $\mathrm{ind}(\phi(y_1, y_2, y_3))$ with respect to y_1. The positivity set for ϕ satisfies

$$1 - y_1^2 - (y_2 - y_3 - 1)^2 > 0 \;\Leftrightarrow\; |y_1| < \sqrt{1 - (y_2 - y_3 - 1)^2},$$

provided that $|y_2 - y_3 - 1| < 1 \Leftrightarrow 0 < y_2 - y_3 < 2$. Thus

$$(P_1 f_t)(y_2, y_3) \;=\; \begin{cases} 0 & \text{if } |y_2 - y_3 - 1| \geq 1 \\ \displaystyle\int_{-\sqrt{1-(y_2-y_3-1)^2}}^{\sqrt{1-(y_2-y_3-1)^2}} \rho(y_1)\,\mathrm{d}y_1 & \text{if } 0 < y_2 - y_3 < 2, \end{cases} \tag{16}$$

where ρ is the standard normal density. Expanding $\rho(y_1)$ around 0, we have for $y_2 - y_3$ small and positive that

$$(P_1 f_t)(y_2, y_3) \;\approx\; 2\rho(0)\sqrt{1 - (y_2 - y_3 - 1)^2} \;\approx\; 2\rho(0)\sqrt{2(y_2 - y_3)}. \tag{17}$$

Thus there is a square-root singularity at $(y_2, y_3) = (0, 0)$ in any direction within the half plane $y_2 > y_3$. Note that, in accordance with Theorem 1, this is exactly the set of directions $\boldsymbol{z}$ which have a positive inner product with $\nabla_{-1}\phi(\boldsymbol{y}^*)$.

Finally, we come to consider the mixed partial derivatives of $P_1 f_t$, to see if that function could be covered by standard QMC or SG error analysis. It is of course possible to compute any mixed derivatives of $P_1 f$ using the expression (16), but the resulting expressions are inevitably complicated. It is more instructive to study as a surrogate the approximation given by (17). Thus ignoring the multiplying constants we define

$$g(y_2, y_3) \;:=\; \sqrt{y_2 - y_3} \qquad \text{for} \quad 0 < y_2 - y_3 < 2.$$

Then trivially we have

$$\frac{\partial g}{\partial y_2}(y_2, y_3) = \frac{1}{2\sqrt{y_2 - y_3}} \quad \text{and} \quad \frac{\partial^2 g}{\partial y_3 \, \partial y_2}(y_2, y_3) = -\frac{1}{4}(y_2 - y_3)^{-3/2}.$$

Since it is easily seen that the latter expression is not integrable over the domain $0 < y_2 - y_3 < 2$, it is manifest that g does not belong to any mixed derivative space of integer order. Nor, similarly, does $P_1 f_t$ belong to any such space.

As a real-world example, consider the displacement of a cantilever beam under vertical and horizontal loads from [4] and which has also been studied in [16].

Example 6 The displacement of a cantilever beam is a random variable X given by

$$X = \phi(\boldsymbol{Y}) = \frac{4\ell^3}{\omega\tau Y_3}\sqrt{\frac{Y_1^2}{\omega^4} + \frac{Y_2^2}{\tau^4}}, \tag{18}$$

where Y_1 is a random variable for the horizontal load, Y_2 is a random variable for the vertical load and Y_3 is a random variable for Young's modulus. For simplicity, we assume each Y_i is an independent standard normal random variable, $Y_i \sim \mathrm{N}(0, 1)$, and $\ell = 100$, $\omega = 4$, $\tau = 2$ are constants.

The probability that the displacement is greater than $t \in \mathbb{R}$ is $\mathbb{P}[X \geq t] = \mathbb{E}[\mathrm{ind}(X - t)]$, which is given by the integral (1) with $f = f_t$ of the form (9) and hence fits into the setting of this paper. In a related direction, in [16, Sect. 4.2] L'Ecuyer and colleagues approximated the cumulative distribution function, $F(t) = \mathbb{P}[X \leq t] = 1 - \mathbb{P}[X \geq t]$, using randomized QMC after first performing preintegration (referred to in that paper as "conditioning"). They separately performed preintegration with respect to each of the three variables in turn, giving an analytic formula for each and also presenting convergence results for subsequently applying a randomized QMC rule. Here we consider the effect of the monotonicity condition and relate this back to their results. Note that to be consistent with our notation, we have relabelled the variables compared to [16] and we now treat ϕ in (18) as function of the deterministic variable $\boldsymbol{y} \in \mathbb{R}^3$.

Consider preintegration with respect to y_1. At $\boldsymbol{y}^* = (0, 1, 1)$, we have

$$\frac{\partial \phi}{\partial y_1}(\boldsymbol{y}^*) = 0, \quad \frac{\partial^2 \phi}{\partial y_1^2}(\boldsymbol{y}^*) = \frac{4\ell^3\tau}{\omega^5} > 0, \quad \nabla\phi(\boldsymbol{y}^*) = \frac{4\ell^3}{\omega\tau^3}\begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix} \neq \boldsymbol{0}. \tag{19}$$

Hence, condition (12) is satisfied and Theorem 1 implies that $P_1 f_t$ will have a square-root singularity for $t = \phi(\boldsymbol{y}^*) = 4\ell^3/(\omega\tau^3)$ at $\boldsymbol{y}^* = (0, 1, 1)$. (Although $\phi \notin C^2(\mathbb{R}^3)$, it is C^2 in a neighborhood of $\boldsymbol{y}^*$, and so Theorem 1 still holds as argued in Remark 2.) Indeed, in this case $P_1 f_t$ is given by

$$(P_1 f_t)(y_2, y_3) \;=\; 1 - \int_{-\frac{\omega^2}{\tau^2}\sqrt{y_3^2-y_2^2}}^{\frac{\omega^2}{\tau^2}\sqrt{y_3^2-y_2^2}} \rho(y_1)\,\mathrm{d}y_1 \;\approx\; 1 - \frac{2\omega^2}{\tau^2}\rho(0)\sqrt{y_3^2 - y_2^2}\,,$$

where again ρ is the standard normal normal density. Thus, $P_1 f_t$ has a square-root singularity at $y_2 = y_3 = 1$.

From Theorem 1 one might suspect, because t in the theorem has the particular value $t = \phi(\boldsymbol{y}^*)$, that singularities of this kind are rare. However, in the following theorem we show that values of $t \in \mathbb{R}$ at which singularities occur in $P_1 f_t$ are often not isolated. This is essentially because points at which $(\partial\phi/\partial y_1)(\boldsymbol{y}) = 0$ are themselves not isolated.

Theorem 2 *Let $\phi \in C^2(\mathbb{R}^d)$, $\rho \in C(\mathbb{R})$ and $\mathrm{supp}(\rho) = \mathbb{R}$, and assume that $\boldsymbol{y}^* \in \mathbb{R}^d$ is such that*

$$\frac{\partial\phi}{\partial y_1}(\boldsymbol{y}^*) = 0, \quad \nabla\phi(\boldsymbol{y}^*) \neq \boldsymbol{0}, \quad \textit{and} \quad \nabla\frac{\partial\phi}{\partial y_1}(\boldsymbol{y}^*) \neq \boldsymbol{0}, \tag{20}$$

with $\nabla\phi(\boldsymbol{y}^*)$ not parallel to $\nabla(\partial\phi/\partial y_1)(\boldsymbol{y}^*)$. Then for any t in some open interval containing $\phi(\boldsymbol{y}^*)$, there exists a point $\boldsymbol{y}^{(t)} \in \mathbb{R}^d$ in a neighborhood of $\boldsymbol{y}^*$ at which

$$\phi(\boldsymbol{y}^{(t)}) = t, \quad \frac{\partial\phi}{\partial y_1}(\boldsymbol{y}^{(t)}) = 0, \quad \textit{and} \quad \nabla\phi(\boldsymbol{y}^{(t)}) \neq \boldsymbol{0}. \tag{21}$$

Moreover, if we assume also that $(\partial^2\phi/\partial y_1^2)(\boldsymbol{y}^*) \neq 0$, then the function $(P_1 f_t)(\boldsymbol{y}_{-1})$ as defined in (10) has a square-root singularity at $\boldsymbol{y}_{-1}^{(t)} \in \mathbb{R}^{d-1}$ along any line in $\mathbb{R}^{d-1}$ through $\boldsymbol{y}_{-1}^{(t)}$ in any direction in $\mathbb{R}^{d-1}$ that has a positive inner product with $\nabla_{-1}\phi(\boldsymbol{y}^{(t)})$.

Proof It is convenient to define $\psi(\boldsymbol{y}) := (\partial\phi/\partial y_1)(\boldsymbol{y})$, which by assumption is a real-valued $C^1(\mathbb{R}^d)$ function that satisfies

$$\psi(\boldsymbol{y}^*) = 0 \quad \text{and} \quad \nabla\psi(\boldsymbol{y}^*) \neq \boldsymbol{0}.$$

We need to show that for t in some open interval containing $\phi(\boldsymbol{y}^*)$ there exists $\boldsymbol{y}^{(t)} \in \mathbb{R}^d$ in a neighborhood of $\boldsymbol{y}^*$ at which

$$\phi(\boldsymbol{y}^{(t)}) = t, \quad \psi(\boldsymbol{y}^{(t)}) = 0, \quad \text{and} \quad \nabla\phi(\boldsymbol{y}^{(t)}) \neq \boldsymbol{0}.$$

Clearly, we can confine our search for $\boldsymbol{y}^{(t)}$ to the zero level set of ψ, that is, to the solutions of

$$\psi(\boldsymbol{y}) = 0, \quad \boldsymbol{y} \in \mathbb{R}^d.$$

Since $\nabla\psi(\boldsymbol{y})$ is continuous and non-zero in a neighborhood of $\boldsymbol{y}^*$, the zero level set of ψ is a manifold of dimension $d-1$ near $\boldsymbol{y}^*$, whose tangent hyperplane at $\boldsymbol{y}^*$ is orthogonal to $\nabla\psi(\boldsymbol{y}^*)$, see, e.g., [17, Chap. 5]. On this hyperplane, there is a search

direction starting from $\boldsymbol{y}^*$ for which $\phi(\boldsymbol{y})$ has a maximal rate of increase, namely the direction of the orthogonal projection of $\nabla\phi(\boldsymbol{y}^*)$ onto the tangent hyperplane. This is a non-zero vector because $\nabla\phi(\boldsymbol{y}^*)$ is not parallel to $\nabla\psi(\boldsymbol{y}^*)$. Setting out from the point $\boldsymbol{y}^*$ in the direction of positive gradient, the value of ϕ is strictly increasing in a sufficiently small neighborhood of $\boldsymbol{y}^*$, while in the direction of negative gradient it is strictly decreasing. Thus searching on the manifold for a $\boldsymbol{y}^{(t)}$ such that $\phi(\boldsymbol{y}^{(t)}) = t$ will be successful in one of these directions for t in a sufficiently small open interval containing $\phi(\boldsymbol{y}^*)$.

Under the additional assumption that $(\partial^2\phi/\partial y_1^2)(\boldsymbol{y}^*) \neq 0$, all the conditions of Theorem 1 are satisfied with $\boldsymbol{y}^*$ replaced by $\boldsymbol{y}^{(t)}$, noting that because $\phi \in C^2(\mathbb{R}^d)$, the second derivative is also non-zero in a sufficiently small neighborhood of $\boldsymbol{y}^*$. This completes the proof. □

Remark 4 We now show that for ϕ as in Examples 1–3 and 5, the singularities in $P_1 f_t$, with f_t as in (9), are not isolated, and accord with Theorem 2. For Example 4, Theorem 2 is not applicable.

- For ϕ as in Example 1 we may choose $\boldsymbol{y}^* = (0, 0)$, as in Remark 3. Indeed, the gradient of the first derivative with respect to y_1 is $\nabla(\partial\phi/\partial y_1)(y_1, y_2) = (-2, 0)$, which is not parallel to $\nabla\phi(y_1, y_2) = (-2y_1, 1)$ for all $(y_1, y_2) \in \mathbb{R}^2$. It follows that (20) holds, e.g., at $\boldsymbol{y}^* = (0, 0)$. Hence Theorem 2 implies that for t in some interval around $\phi(\boldsymbol{y}^*) = 0$ there is $\boldsymbol{y}^{(t)} = (y_1^{(t)}, y_2^{(t)})$ in a neighborhood of $\boldsymbol{y}^* = (0, 0)$ such that $\phi(\boldsymbol{y}^{(t)}) = t$ and (21) holds. In particular, there is still a square-root singularity in $(P_1 f_t)(y_2)$ at $y_2 = y_2^{(t)}$. We confirm that this is indeed the case by taking $\boldsymbol{y}^{(t)} = (0, t)$ and by observing that, as is easily verified, $(P_1 f_t)(y_2)$ has a square-root singularity at $y_2 = t$ for all real numbers t. This singularity in $P_1 f_t$ is similar to the singularity depicted in Fig. 1, but translated by t.
- For ϕ as in Example 3 we can consider $\boldsymbol{y}^* = (0, \pm 1)$ as in Remark 3. The gradient of the first derivative with respect to y_1 is $\nabla(\partial\phi/\partial y_1)(y_1, y_2) = (-2, 0)$, which is not parallel to $\nabla\phi(y_1, y_2) = (-2y_1, 2y_2)$ for all $(y_1, y_2) \in \mathbb{R}^2$ with $y_2 \neq 0$. Thus (20) holds, e.g., at $\boldsymbol{y}^* = (0, \pm 1)$. Theorem 2 implies that for t in some interval around $\phi(\boldsymbol{y}^*) = 1$ there is a point $\boldsymbol{y}^{(t)}$ in a neighborhood of $\boldsymbol{y}^* = (0, \pm 1)$ such that $\phi(\boldsymbol{y}^{(t)}) = t$, (21) holds, and $(P_1 f_t)(y_2)$ has a square-root singularity at $y_2 = y_2^{(t)}$. Indeed, for any real number $t > 0$, taking $\boldsymbol{y}^{(t)} = (0, \pm\sqrt{t})$ gives $\phi(\boldsymbol{y}^{(t)}) = t$, and it can easily be verified that $(P_1 f_t)(y_2)$ has two square-root singularities at $y_2 = \pm\sqrt{t}$. In this case the behavior of $P_1 f_t$ is similar to Fig. 2, with the location of the singularities now depending on t.
- Since ϕ from Example 2 is simply a translation of Example 3 by -1, similar singularities exist for that case for $t > -1$.
- For ϕ as in Example 4 the condition (20) never holds, since $\nabla(\partial\phi/\partial y_1)(y_1, y_2) = (0, 0)$ whenever $(\partial\phi/\partial y_1)(y_1, y_2) = 0$. So no conclusion can be drawn from Theorem 2 in this case. It is no contradiction that, as is easily seen, there is a singularity (of cube-root character) in $(P_1 f_t)(y_2)$ at $y_2 = t$ for every real number t.
- For ϕ as in Example 5 the condition (20) is satisfied for $\boldsymbol{y}^* = (0, 0, 0)$, with the two gradients not parallel, thus Theorem 2 is applicable. It is easily seen that the point $(0, y_2^{(t)}, y_3^{(t)})$ satisfies (21) for every $t < 1$ provided $y_2^{(t)} - y_3^{(t)} = 1 \pm \sqrt{1 - t}$.

- For ϕ giving the displacement of a cantilever beam as in Example 6, from the calculation (19) it follows that condition (20) is satisfied for $\boldsymbol{y}^* = (0, 1, 1)$ and also that the gradients $\nabla\phi(\boldsymbol{y}^*)$ and $\nabla(\partial\phi/\partial y_1)(\boldsymbol{y}^*)$ are not parallel. Thus, Theorem 2 applies. It is easily seen that, for all $t > 0$, condition (21) is satisfied for $(0, y_2^{(t)}, y_3^{(t)})$ with $y_2^{(t)} = t$ and $y_3^{(t)} = 4\ell^3/(\omega\tau^3)$.

3 A High-Dimensional Example

Motivated by applications in computational finance, for a high-dimensional example we consider the problem of approximating the fair price for a *digital Asian option*, a problem that can be formulated as an integral as in (1) with a discontinuous integrand of the form (9). When monotonicity holds, it was shown in [12] that preintegration not only has theoretical smoothing benefits, but also that when followed by a QMC rule to compute the $(d-1)$-dimensional integral the computational experience can be excellent. On the other hand, that paper provided no insight as to what happens, either theoretically or numerically, when monotonicity fails. In this section, in contrast, we will deliberately apply preintegration using a chosen variable for which the monotonicity condition fails. We will demonstrate the resulting lack of smoothness, using the theoretical results from the previous section, and show that the performance of the subsequent QMC rule can degrade when the preintegration variable lacks the monotonicity property.

For a given strike price K, the payoff for a digital Asian call option is given by

$$\text{payoff} \;=\; \text{ind}(\phi - K),$$

where ϕ is the average price of the underlying stock over the time period. Under the Black–Scholes model, the time-discretized average is given by

$$\phi(\boldsymbol{y}) \;=\; \frac{S_0}{d}\sum_{k=1}^{d}\exp\left((r-\tfrac{1}{2}\sigma^2)\frac{kT}{d}+\sigma\boldsymbol{A}_k\boldsymbol{y}\right), \tag{22}$$

where $\boldsymbol{y} = (y_k)_{k=1}^d$ is a vector of i.i.d. standard normal random variables, S_0 is the initial stock price, T is the final time, r is the risk-free interest rate, σ is the volatility and d is the number of time steps, which is also the dimension of the problem. Note that in (22) we have already made a change of variables to write the problem in terms of standard normal random variables, by factorizing the covariance matrix of the Brownian motion as $\Sigma = AA^\top$, where the entries of the covariance matrix are $\Sigma_{k,\ell} = \min(k,\ell)\times T/d$. Then in (22), $\boldsymbol{A}_k$ denotes the kth row of this matrix factor.

The fair price of the option is then given by the discounted expected payoff

$$e^{-rT}\,\mathbb{E}[\text{payoff}] \;=\; e^{-rT}\int_{\mathbb{R}^d}\text{ind}(\phi(\boldsymbol{y})-K)\rho_d(\boldsymbol{y})\,\mathrm{d}\boldsymbol{y}\,. \tag{23}$$

Letting $f(\boldsymbol{y}) = \mathrm{ind}(\phi(\boldsymbol{y}) - K)$, this example clearly fits into the framework (9), where ϕ is the average stock price (22), t takes the value of the strike price K and each ρ is a standard normal density.

There are three popular methods for factorizing the covariance matrix: the *standard construction* (which uses the Cholesky factorization), *Brownian bridge*, and *principal components* or PCA, see, e.g., [8] for further details. In the first two cases, all components of the matrix A are positive, in which case it is easily seen by studying the derivative of (22) with respect to y_j for some $j = 1, \ldots, d$,

$$\frac{\partial \phi}{\partial y_j}(\boldsymbol{y}) \;=\; \frac{S_0}{d} \sum_{k=1}^{d} \sigma A_{k,j} \exp\left((r - \tfrac{1}{2}\sigma^2) \frac{kT}{d} + \sigma \boldsymbol{A}_k \boldsymbol{y} \right),$$

that ϕ is monotone increasing with respect to y_j no matter which j is chosen.

In contrast, for the PCA construction, which we consider below, the situation is very different, in that there is only one choice of j for which ϕ is monotone with respect to y_j. This is because with PCA the factorization of Σ employs its eigendecomposition, with the jth column of A being a (scaled) eigenvector corresponding to the jth eigenvalue labeled in decreasing order. Since the covariance matrix Σ has all entries positive, the eigenvector corresponding to the largest eigenvalue can be scaled to have all components positive. Thus, for $j = 1$ monotonicity of ϕ is achieved. On the other hand, every eigenvector other than the first is orthogonal to the first, and therefore must have components of both signs. Given that

$$A_{k,j} > 0 \implies \exp(A_{k,j} y_j) \to \begin{cases} +\infty & \text{as } y_j \to +\infty, \\ 0 & \text{as } y_j \to -\infty, \end{cases}$$

$$A_{k,j} < 0 \implies \exp(A_{k,j} y_j) \to \begin{cases} 0 & \text{as } y_j \to +\infty, \\ +\infty & \text{as } y_j \to -\infty, \end{cases}$$

it follows that for $j \neq 1$ there is at least one term in the sum over k in (22) that approaches $+\infty$ as $y_j \to +\infty$ and at least one other term that approaches $+\infty$ as $y_j \to -\infty$. Given that all terms in the sum over k in (22) are positive, it follows that for the PCA case and $j \neq 1$, ϕ must approach $+\infty$ as $y_j \to \pm\infty$, so is definitely not monotone. Moreover, with respect to each variable y_j the function ϕ is strictly convex, since

$$\frac{\partial^2 \phi}{\partial y_j^2}(\boldsymbol{y}) \;=\; \frac{S_0}{d} \sum_{k=1}^{d} (\sigma A_{k,j})^2 \exp\left((r - \tfrac{1}{2}\sigma^2) \frac{kT}{d} + \sigma \boldsymbol{A}_k \boldsymbol{y} \right) \;>\; 0 \quad \text{for all } \boldsymbol{y} \in \mathbb{R}^d.$$

For definiteness, in the following discussion we denote by y_2 the preintegration variable for which monotonicity fails, and denote the other variables by $\boldsymbol{y}_{-2} = (y_1, y_3, \ldots, y_d)$. We now use the results from the previous section to determine the smoothness, or rather the lack thereof, of $P_2 f$ when $f(\boldsymbol{y}) = \mathrm{ind}(\phi(\boldsymbol{y}) - K)$. To do

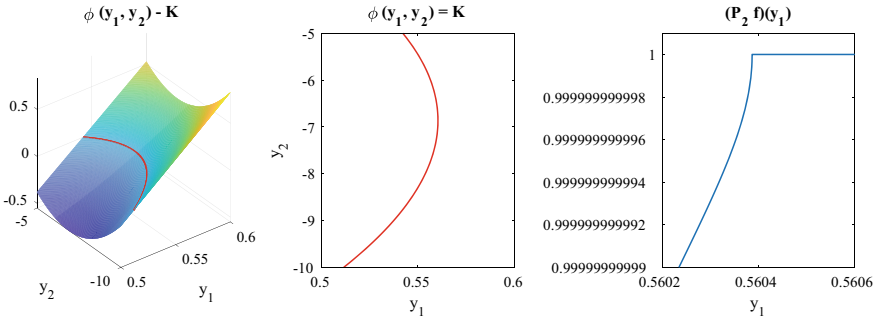


Fig. 5 Illustrations for digital Asian option in two dimensions

this we will use Theorems 1 and 2, where with a slight abuse of notation we replace y_1 by y_2 as our special preintegration variable.

First, note that we have already established that ϕ is not monotone with respect to y_2, and since ϕ is strictly convex with respect to y_2, for a given $\boldsymbol{y}_{-2}$, there exists a unique $y_2^* \in \mathbb{R}$ such that $(\partial\phi/\partial y_2)(y_2^*, \boldsymbol{y}_{-2}) = 0$. Since ϕ is strictly increasing with respect to y_1, it follows that $\nabla\phi(y_2^*, \boldsymbol{y}_{-2}) \neq \boldsymbol{0}$. Furthermore, since $(\partial^2\phi/\partial y_2^2)(y_2^*, \boldsymbol{y}_{-2}) > 0$, Theorem 1 implies that for $K = \phi(y_2^*, \boldsymbol{y}_{-2})$, the preintegrated function $P_2 f$ has a square-root singularity along any line not orthogonal to $\nabla_{-2}\phi(y_2^*, \boldsymbol{y}_{-2})$, with ∇_{-2} defined analogously to ∇_{-1} in Theorem 1.

Furthermore, Theorem 2 implies that this singularity is not isolated. To apply Theorem 2, we note that we have already established the first two conditions in (20) (recall again that we now take y_2 as the preintegration variable). We also have $(\partial^2\phi/\partial y_2^2)(y_2^*, \boldsymbol{y}_{-2}) > 0$, which implies $\nabla(\partial\phi/\partial y_2)(y_2^*, \boldsymbol{y}_{-2}) \neq \boldsymbol{0}$. Moreover, we know that $\nabla(\partial\phi/\partial y_2)(y_2^*, \boldsymbol{y}_{-2})$ and $\nabla\phi(y_2^*, \boldsymbol{y}_{-2})$ are not parallel, since the former has a positive second component while the latter has a zero second component.

To visualize this singularity, in Fig. 5 we provide an illustration of the option in two dimensions. (Note that we consider $d = 2$ here for visualiZation purposes only; we have shown already that the singularity exists for any choice of $d > 1$. Later we present numerical results for $d = 256$.) Figure 5 gives a contour plot of $\phi(y_1, y_2) - K$ (left), the zero level set of $\phi(y_1, y_2) = K$ (middle) and then the graph of $P_2 f$ (right). As expected, we can clearly see that $P_2 f$ has a singularity that is of square-root nature.

To perform the preintegration step $P_2 f$ in practice, note that since ϕ is strictly convex with respect to y_2, for each $\boldsymbol{y}_{-2} \in \mathbb{R}^{d-1}$, there is a single turning point $y_2^* \in \mathbb{R}$ for which $(\partial\phi/\partial y_2)(y_2^*, \boldsymbol{y}_{-2}) = 0$ and $\phi(y_2^*, \boldsymbol{y}_{-2})$ is a global minimum. It follows that there are at most *two* distinct points, $\xi_a(\boldsymbol{y}_{-2}) \leq \xi_b(\boldsymbol{y}_{-2})$, such that

$$\phi(\xi_a(\boldsymbol{y}_{-2}), \boldsymbol{y}_{-2}) \;=\; K \;=\; \phi(\xi_b(\boldsymbol{y}_{-2}), \boldsymbol{y}_{-2})\,.$$

Preintegration with respect to y_2 then simplifies to

$$
\begin{aligned}
(P_2 f)(\boldsymbol{y}_{-2}) &= \int_{-\infty}^{\infty} \operatorname{ind}(\phi(y_2, \boldsymbol{y}_{-2}) - K)\, \rho(y_2)\, \mathrm{d}y_2 \\
&= \begin{cases} 1 & \text{if } \phi(y_2^*, \boldsymbol{y}_{-2}) \geq K, \\ \displaystyle\int_{-\infty}^{\xi_a(\boldsymbol{y}_{-2})} \rho(y_2)\, \mathrm{d}y_2 + \int_{\xi_b(\boldsymbol{y}_{-2})}^{\infty} \rho(y_2)\, \mathrm{d}y_2 & \text{otherwise,} \end{cases} \\
&= \begin{cases} 1 & \text{if } \phi(y_2^*, \boldsymbol{y}_{-2}) \geq K, \\ \Phi(\xi_a(\boldsymbol{y}_{-2})) + 1 - \Phi(\xi_b(\boldsymbol{y}_{-2})) & \text{otherwise.} \end{cases}
\end{aligned}
$$

In practice, for each $\boldsymbol{y}_{-2} \in \mathbb{R}^{d-1}$ the turning point y_2^* and the points of discontinuity $\xi_a(\boldsymbol{y}_{-2})$ and $\xi_b(\boldsymbol{y}_{-2})$ are computed numerically, e.g., we use Newton's method in our numerical experiments.

We now look at how this lack of smoothness affects the performance of using a numerical preintegration method to approximate the fair price for the digital Asian option in $d = 256$ dimensions. Explicitly, we approximate the integral in (23) by applying a $(d-1)$-dimensional QMC rule to $P_2 f$. As a comparison, we also present results for approximating the integral in (23) by applying the same QMC rule to $P_1 f$. Recall that ϕ is monotone in dimension 1 and, furthermore, it was shown in [12] that $P_1 f$ is smooth.

For the QMC rule we use a randomly shifted lattice rule based on the generating vector `lattice-32001-1024-1048576.3600` from [14] using $N = 2^{10}, 2^{11}, \ldots, 2^{19}$ points. For each N, the final approximation is the average of 100 shifted QMC approximations corresponding to $R = 100$ independent random shifts. The RMSE is then estimated using the sample standard error over the R independent shifted approximations in the usual way. The parameters for the option are $S_0 = \$100$, $K = \$110$, $T = 1$, $d = 256$ timesteps, $r = 0.1$ and $\sigma = 0.1$. We also performed a standard Monte Carlo approximation using $R \times N$ points and a plain (without preintegration) QMC approximation using the same generating vector. All methods use the PCA factorization in the formulation (22), so that the integrand is consistent across the different experiments.

In Fig. 6, we plot the convergence of the standard error in terms of the total number of function evaluations $R \times N$. We can clearly see that preintegration with respect to y_2 produces less accurate results compared to preintegration with respect to y_1, with errors that are up to an order of magnitude larger for the higher values of N. We also note that to achieve a given error, say 10^{-4}, the number of points needs to be increased tenfold. Furthermore, we observe that the empirical convergence rate for preintegration with respect to y_2 is $N^{-0.9}$, which is sightly worse than the rate of $N^{-0.97}$ for preintegration with respect to y_1. Hence, when the monotonicity condition fails, not only does the theory for QMC fail due to the presence of a singularity, but we also observe worse results in practice and somewhat slower convergence.

We also plot the error of standard Monte Carlo and QMC approximations, which behave as expected and are both significantly outperformed by the two preintegration methods.

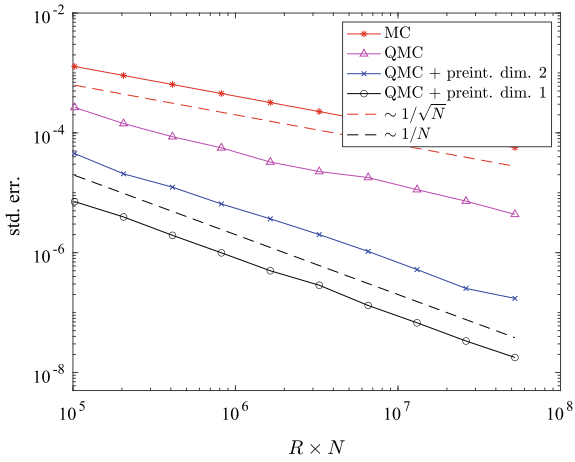


Fig. 6 Convergence in N for different approximations of the fair price for a digital Asian option

4 Conclusion

If the monotonicity property (11) fails and $f = f_t$ is defined by (9) then we have seen that generically there is a singularity in $P_1 f$ for some values of t, and under known conditions this is true even for all values of t in an interval.

It should also be noted that the implementation of preintegration is more difficult if monotonicity fails, since instead of a single integral from $\xi(\boldsymbol{y}_{-1})$ to ∞, as in (7), there will in general be additional finite or infinite intervals to integrate over, all of whose end points must be discovered by the user for each required value of $\boldsymbol{y}_{-1}$.

To explore the consequences of a lack of monotonicity empirically, we carried out in Sect. 3 a 256-dimensional calculation of pricing a digital Asian option, first by preintegrating with respect to a variable known to lack the monotonicity property, and then with a variable where the property holds, with the result that both accuracy and rate of convergence were observed to be degraded when monotonicity fails.

There is an additional problem of preintegrating with respect to a variable for which the monotonicity fails, namely that because of the proven lack of smoothness, the resulting preintegrated function no longer belongs to the space of $(d-1)$-variate functions of dominating mixed smoothness of order one. As a consequence, there is at present no theoretical support for the use of QMC integration for this $(d-1)$-dimensional integral.

The practical significance of this paper is that effective use of preintegration is greatly enhanced by the preliminary identification of a special variable for which the monotonicity property is known to hold. The paper does not offer guidance on the choice of variable if there is more than one such variable. In such cases it may be natural to choose the variable for which preintegration leads to the greatest reduction in variance, see [16, Sect. 2.4] for related work.

Acknowledgements The authors acknowledge the support of the Australian Research Council under the Discovery Project DP210100831. They also wish to acknowledge the inspirational contributions of Pierre L'Ecuyer to random number generation and the world of Quasi-Monte Carlo.

References

1. Achtsis, N., Cools, R., Nuyens, D.: Conditional sampling for barrier option pricing under the LT method. SIAM J. Financ. Math. **4**, 327–352 (2013)
2. Achtsis, N., Cools, R., Nuyens, D.: Conditional sampling for barrier option pricing under the Heston model. In: Dick, J., Kuo, F.Y., Peters, G.W., Sloan, I.H. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2012, pp. 253–269. Springer, Berlin/Heidelberg (2013)
3. Bayer, C., Siebenmorgen, M., Tempone, R.: Smoothing the payoff for efficient computation of Basket option prices. Quant. Finance **18**, 491–505 (2018)
4. Bingham, D.: Virtual library of simulation experiments (2017). https://www.sfu.ca/~ssurjano/canti.html
5. Bungartz, H., Griebel, M.: Sparse grids. Acta Numer. **13**, 147–269 (2004)
6. Dick, J., Kuo, F.Y., Sloan, I.H.: High dimensional integration - the quasi-Monte Carlo way. Acta Numer. **22**, 133–288 (2013)
7. Gilbert, A.D., Kuo, F.Y., Sloan, I.H.: Analysis of preintegration followed by quasi-Monte Carlo integration for distribution functions and densities (2021). arXiv:2112.10308
8. Glasserman, P.: Monte Carlo Methods in Financial Engineering. Springer, Berlin (2003)
9. Glasserman, P., Staum, J.: Conditioning on one-step survival for barrier option simulations. Oper. Res. **49**, 923–937 (2001)
10. Griebel, M., Kuo, F.Y., Sloan, I.H.: The smoothing effect of integration in $\mathbb{R}^d$ and the ANOVA decomposition. Math. Comp. **82**, 383–400 (2013)
11. Griebel, M., Kuo, F.Y., Sloan, I.H.: Note on "the smoothing effect of integration in $\mathbb{R}^d$ and the ANOVA decomposition." Math. Comp. **86**, 1847–1854 (2017)
12. Griewank, A., Kuo, F.Y., Leövey, H., Sloan, I.H.: High dimensional integration of kinks and jumps - smoothing by preintegration. J. Comput. Appl. Math. **344**, 259–274 (2018)
13. Holtz, M.: Sparse grid quadrature in high dimensions with applications in finance and insurance (PhD thesis). Springer, Berlin (2011)
14. Kuo, F.Y.: (2007). https://web.maths.unsw.edu.au/~fkuo/lattice/index.html. Accessed 13 Dec 2021
15. L'Ecuyer, P., Puchhammer, F.: Density estimation by Monte Carlo and quasi-Monte Carlo (2021). arXiv:2103.15976
16. L'Ecuyer, P., Puchhammer, F., Ben Abdellah, A.: Monte Carlo and quasi-Monte Carlo density estimation via conditioning (2021). arXiv:1906.04607
17. Lee, J.M.: Introduction to Smooth Manifolds. Springer, New York (2000)
18. Nuyens, D., Waterhouse, B.J.: A global adaptive quasi-Monte Carlo algorithm for functions of low truncation dimension applied to problems from finance. In: Plaskota, L., Woźniakowski, H. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2010, pp. 589–607. Springer, Berlin/Heidelberg (2012)
19. Weng, C., Wang, X., He, Z.: Efficient computation of option prices and greeks by quasi-Monte Carlo method with smoothing and dimension reduction. SIAM J. Sci. Comput. **39**, B298–B322 (2017)

Combined Derivative Estimators

Paul Glasserman

Abstract We discuss combinations of simulation-based derivative estimators using infinitesimal perturbation analysis (IPA) and the likelihood ratio method (LRM). We first provide a historical perspective on combinations of IPA and LRM and then turn to connections with the generalized likelihood ratio (GLR) method. We re-derive a GLR estimator for barrier options through a combination of IPA and LRM. We then consider the behavior of a GLR estimator for a discrete-time approximation to a diffusion process as the time step shrinks. We show that an average of low-rank GLR estimators has a continuous-time limit, even though each individual estimator blows up. The limit matches an estimator previously derived through Malliavin calculus and also through a combination of IPA and LRM.

Keywords Sensitivity analysis · Simulation · Likelihood ratio method

1 Introduction

I first met Pierre in 1987, at the ORSA/TIMS conference in St. Louis. I was in my last year of graduate school, attending a major conference for the first time, and presenting something on derivative estimation. I was pleasantly surprised by Pierre's friendly manner toward a random student; but I was even more surprised to learn of his own ongoing work on derivative estimation, along with other topics in simulation and dynamic programming. I have continued to value Pierre's collegiality and openness, his broad research interests and accomplishments, his work in fostering a community of scholars, and our shared interest in derivative estimation for the past 34 years.

Several of Pierre's papers have involved combinations, comparisons, and extensions of derivative estimators. I will build on this theme and discuss some combinations of infinitesimal perturbation analysis (IPA) and likelihood ratio method (LRM) estimators. Pierre's most recent work on derivative estimation considers the generalized likelihood ratio (GLR) method introduced in Peng et al. [23]. I will discuss

P. Glasserman (✉)
Columbia Business School, New York, NY, USA
e-mail: pg20@columbia.edu

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_10

some examples relating GLR estimators to combinations of IPA and LRM estimators. This paper's main observation concerns an application of GLR in the setting of a time-discretized diffusion process: averaging multiple low-rank GLR estimators leads to a method that works in the continuous-time limit, even though each individual estimator blows up in the limit. This result uses a connection between the GLR estimator and a combined IPA-LRM estimator. It also suggests a connection between GLR and estimators derived using Malliavin calculus.

2 Derivative Estimation

This section provides some background on derivative estimation and discusses some of Pierre's contribution to the topic, with particular emphasis on combinations of methods.

2.1 Background

Let $V(\theta)$ denote the output of a stochastic simulation depending on a parameters θ. In addition to estimating the expectation $v(\theta) = \mathbb{E}[V(\theta)]$, we may be interested in estimating the derivative $v'(\theta)$ for purposes of sensitivity analysis or optimization. If the interchange of derivative and expectation in $v'(\theta) = \mathbb{E}[V'(\theta)]$ is valid, then $V'(\theta)$ provides an unbiased estimator of $v'(\theta)$. Here, $V'(\theta)$ should be understood as the derivative of the simulation output with respect to θ, holding all other inputs fixed. This is the basis for infinitesimal perturbation analysis (IPA) derivative estimation, introduced in Ho and Cao [16] and Zazanis and Suri [31].

An alternative approach starts from a representation of the form

$$v(\theta) = \int \nu f(\nu, \theta)\, d\nu,$$

where $f(\cdot, \theta)$ is the probability density of $V(\theta)$. Differentiating under the integral leads to

$$v'(\theta) = \int \nu \partial_\theta f(\nu, \theta)\, d\nu = \int \nu \frac{\partial_\theta f(\nu, \theta)}{f(\nu, \theta)} f(\nu, \theta)\, d\nu = \mathbb{E}_\theta[V \cdot s_\theta(V)], \qquad (1)$$

where $\mathbb{E}_\theta$ denotes expectation with respect to $f(\cdot, \theta)$, and

$$s_\theta(\nu) = \partial_\theta \log f(\nu, \theta)$$

is often called a score function. When the steps in this derivation are valid, $V \cdot s_\theta(V)$ provides an unbiased estimator of $v'(\theta)$. This is the likelihood ratio method (LRM) introduced in Glynn [11], Reiman and Weiss [27], and Rubinstein [28].

Putting the parameter in the path $V(\theta)$ leads to the IPA estimator $V'(\theta)$; putting the parameter in the density $f(\cdot, \theta)$ leads to the LRM estimator. We can often move from one representation to another through a change of variables. Rubinstein [29] referred to a change of variables that moves the parameter from the path to the measure as "pushing out."

In practice, the simulation output V is usually a function of input random variables, and θ enters through the distributions of those inputs, in which case the score function is computed from the distributions of the inputs, rather than directly through the distribution of V. In the IPA perspective, this means that $V'(\theta)$ is calculated through the chain rule, differentiating V with respect to the inputs and the inputs with respect to the parameter.

2.2 Combined Estimators

In [17] and [18], Pierre considered expectations of the form

$$v(\theta) = \int g(\nu, \theta) f(\nu, \theta) \, d\nu,$$

with both a functional dependence on the parameter (through g) and a distributional dependence (through f). The arguments used above lead to a representation

$$\begin{aligned} v'(\theta) &= \int \partial_\theta g(\nu, \theta) f(\nu, \theta) \, d\nu + \int g(\nu, \theta) s_\theta(\nu) f(\nu, \theta) \, d\nu \\ &= \mathbb{E}_\theta[\partial_\theta g(V, \theta) + g(V, \theta) s_\theta(V)]. \end{aligned}$$

The result inside the expectation is an estimator that combines an IPA term $\partial_\theta g(V, \theta)$ and an LRM $g(V, \theta) s_\theta(V)$, although Pierre used a slightly different classification.

Both terms require an interchange of derivative and integral; Pierre's papers [17] and [18] gave a unified treatment of the conditions for the interchange. Of course, the general question of when such an interchange is valid is classical. The value of the types of conditions in Pierre's papers and related work lies in their applicability to systems commonly studied through simulation, such as the queueing, reliability, and inventory models analyzed in [17] and [18]. Generally speaking, the interchange conditions for IPA derivatives are more restrictive than for LRM derivatives because commonly used parametric densities tend to be smoother than the sample paths of discrete-event systems and also smoother than the payoffs of option contracts.

2.3 Second Derivatives

In [17], Pierre also considered the estimation of second derivatives. Second derivatives are clearly useful in optimization; they are also routinely calculated in financial applications, where first- and second-order derivatives are used in hedging options.

Combinations of IPA and LRM estimators arise naturally in estimating second derivatives. Repeating the steps leading to (1) yields a pure-LRM estimator of $v''(\theta)$ of the form

$$V \cdot s_\theta^{(2)}(V), \quad s_\theta^{(2)}(v) = \partial_\theta^2 \log f(v,\theta) + (\partial_\theta \log f(v,\theta))^2.$$

In other words, just as multiplying by $s_\theta(V)$ has the effect of differentiating once, multiplying by $s_\theta^{(2)}(V)$ has the effect of differentiating twice.

The pure-IPA estimator $V''(\theta)$ is often uninformative, if it exists at all. Kinks in the paths of discrete-event systems and option payoffs create discontinuities in the first derivative $V'(\theta)$, preventing further differentiation.

However, IPA and LRM can be fruitfully combined. Multiplying the IPA first derivative $V'(\theta)$ by the appropriate score function yields an IPA-LRM second derivative estimator; similarly, taking the pathwise derivative of the LRM estimator (1) yields an LRM-IPA second derivative estimator.

Estimators of this form are derived and compared in the setting of interest rate derivatives in Section 7.3.3 of Glasserman [9]. In that setting, the two hybrid second-derivative estimators perform about equally well and have roughly one-tenth the variance of the pure-LRM second-derivative estimator. The combined estimators appear to benefit from the generally lower variance of IPA-based methods while using LRM to expand their scope beyond first derivatives.

2.4 Finite Difference Estimators and IPA

A rough rule of thumb for the applicability of IPA is that $V(\theta)$ should be continuous throughout an interval of θ values that does not depend on the stochastic inputs to the simulation. Discontinuities in $V(\theta)$ are missed by $V'(\theta)$ but affect $v'(\theta)$.

L'Ecuyer and Perron [21] showed that under conditions ensuring that IPA provides a consistent estimator, a properly implemented finite difference approximation enjoys the same convergence rate. In more detail, let $\bar{V}_n(\theta)$ denote the sample mean of n i.i.d. replications $V_i(\theta)$. Consider a finite difference estimator of the form

$$\Delta_n(\theta) = \frac{\bar{V}_n(\theta + c_n) - \bar{V}_n(\theta)}{c_n}.$$

The replications $V_i(\theta)$ and $V_i(\theta + c_n)$ are implemented using common random numbers, meaning that all inputs other than θ are held fixed. L'Ecuyer and Perron [21]

showed that taking $c_n = O(n^{-1/2})$ ensures that $\Delta_n(\theta)$ estimates $v'(\theta)$ with a root-mean-squared error of $O(n^{-1/2})$. This is the same order as the sample mean $\bar{V}_n'(\theta)$ of IPA derivatives when IPA is unbiased. L'Ecuyer and Perron [21] proved similar results for central difference estimators and steady-state estimation problems. Related results on the convergence rates of finite difference estimators appeared in Glynn [12], Zazanis and Suri [32], and Glasserman and Yao [10].

Based on their comparison results, L'Ecuyer and Perron [21] concluded that IPA, when applicable, can be viewed as an algorithmically efficient implementation of finite differences. The potential efficiency gains are greater when the parameter θ is a vector of dimension d. Applying finite differences to each coordinate requires $d + 1$ separate simulations, and central differences require $2d$ simulations.

The overhead associated with IPA calculations can sometimes be dramatically reduced using "adjoint methods" from the field of algorithmic differentiation (Griewank and Walther [15]). To illustrate, consider a simulation algorithm described through a recursion of the form

$$X_{i+1} = \phi(X_i, Z_{i+1}),$$

where the X_i are d-dimensional, and the Z_i are i.i.d. and vector-valued as well. Suppose the parameter of interest is the initial state x_0, and suppose V is a function of X_n. Differentiating the recursion we get

$$D_{x_0} X_{i+1} = D_x \phi(X_i, Z_{i+1}) \cdot D_{x_0} X_i,$$

where each $D_{x_0} X_i$ and $D_x \phi$ is a $d \times d$ matrix of partial derivatives. The initial matrix $D_{x_0} X_0$ is the identity. The final IPA estimator is $\nabla V \cdot D_{x_0} X_n$.

A forward implementation of this derivative involves n multiplications of $d \times d$ matrices, followed by the vector-matrix product $\nabla V \cdot D_{x_0} X_n$. An adjoint implementation stores the matrices $D_x \phi(X_i, Z_{i+1})$ and then evaluates the product

$$\nabla V \cdot D_x \phi(X_{n-1}, Z_n) \cdots D_x \phi(X_0, Z_1)$$

from left to right, as n vector-matrix products, eliminating the need for any matrix-matrix products.

The adjoint method was introduced to option pricing applications in Giles and Glasserman [7]. The method includes many techniques beyond the simple example given here, and it has found widespread adoption in financial applications; see for example Capriotti and Giles [3]. The computational gains can be dramatic, particularly in calculating sensitivities of a small number of outputs to a large number of parameters — in other words, when the dimension of θ is large relative to the dimension of V.

The adjoint method provides an efficient implementation of IPA, but it does not change the scope of applicability of IPA. In financial applications, the state evolution (the recursion ϕ in the example above) is often continuous, and potential discontinuities are often limited to the evaluation of V once a path has been generated. Giles

[6] applies LRM at the final step to combine an adjoint implementation of IPA with a discontinuous payoff. He shows that applying antithetic variates at the final step can eliminate the leading order error term.

2.5 IPA and Randomized Score Functions

In some cases, LRM estimators can be derived as conditional expectations of IPA estimators. We give a simple example. Suppose U is uniform on [0, 1] and

$$V(\theta) = \mathbf{1}\{U \le p(\theta)\},$$

for some $p(\theta) \in (0, 1)$ varying smoothly with θ. The IPA derivative $V'(\theta)$ equals zero wherever it exists and is therefore uninformative. If we multiply $V(\theta)$ by $|U - p(\theta)|$, we make the dependence on θ continuous because the smoothing factor $|U - p(\theta)|$ equals zero precisely at the discontinuity of V. If we then normalize the smoothing factor by its conditional expectation to leave the original expectation unchanged, we get

$$\tilde{V}(\theta) = \mathbf{1}\{U \le p(\theta)\}\frac{|U - p(\theta)|}{\mathbb{E}[|U - p(\theta)||V(\theta)]} = \mathbf{1}\{U \le p(\theta)\}\frac{p(\theta) - U}{p(\theta)/2}.$$

The normalization ensures that $\mathbb{E}[\tilde{V}(\theta)] = \mathbb{E}[V(\theta)]$, so we can try to estimate $v'(\theta)$ by applying IPA to $\tilde{V}(\theta)$. The IPA derivative is now

$$\tilde{V}'(\theta) = \mathbf{1}\{U \le p(\theta)\}\frac{2p'(\theta)}{p^2(\theta)}U.$$

As $\mathbb{E}[U|V(\theta) = 1] = p(\theta)/2$,

$$\mathbb{E}[\tilde{V}'(\theta)|V(\theta)] = \mathbf{1}\{U \le p(\theta)\}\frac{p'(\theta)}{p(\theta)}.$$

The expression on the right is an LRM estimator because the score function for the Bernoulli random variable $V(\theta)$ is

$$s_\theta(V) = \mathbf{1}\{V = 1\}\frac{p'(\theta)}{p(\theta)} + \mathbf{1}\{V = 0\}\frac{-p'(\theta)}{1 - p(\theta)},$$

and the last term is zero on $\{U \le p(\theta)\}$.

By multiplying the original output by a normalized smoothing factor, applying IPA, and then taking a conditional expectation given the original output variable, we arrived at the LRM estimator for the original problem. Extensions of this idea to

Markov chains and other examples appear in Glasserman [8]. In all such examples, the conditioning step implies that the LRM estimator has lower variance than the IPA estimator; the derivative of the smoothing factor acts like a randomized score function.

2.6 LRM Singularities

A well-known limitation of the likelihood ratio method is that its variance typically increases with the simulation time-horizon. This property makes LRM generally inapplicable with long-run average estimators of steady-state quantities.

L'Ecuyer and Glynn [20] and Glynn and L'Ecuyer [13] addressed this problem by exploiting regenerative structure. When applicable, a regenerative representation expresses a steady-state mean as a ratio of two finite-horizon expectations. A derivative estimator for the ratio can be derived by applying LRM to the numerator and the denominator.

The source of the difficulty in applying LRM over long horizons is that the probability measures over the set of simulation paths at parameters θ and $\theta + h$ may be mutually absolutely continuous for all finite horizons yet mutually singular over an infinite horizon. Once the probability measures are mutually singular, the underpinning for LRM breaks down.

A similar but distinct breakdown can occur in applying LRM to diffusion processes. Such processes are often simulated through discrete-time approximations. The probability measures over paths at different parameter values may be mutually absolutely continuous for all discrete time steps yet mutually singular in the continuous-time limit. We return to this point in Section 4.

2.7 Generalized Likelihood Ratio Method

Pierre's most recent work on derivative estimation has focused on the generalized likelihood ratio (GLR) method, a general framework introduced in Peng et al. [23]. The GLR method is based on representations of the form

$$\partial_\theta \int \nu(g(x,\theta),\theta) f(x,\theta)\, dx = \int \nu(g(x,\theta),\theta) w(x,\theta) f(x,\theta)\, dx, \qquad (2)$$

for a suitable weight function w. Here, the output function ν depends on a transformation g of input variables x. Both ν and g may depend on θ, as may the density f. The weight is derived by first considering sufficiently smooth ν and g for which the derivative can be brought inside the integral, then transforming the derivative of ν with respect to g to a derivative with respect to x, and then applying a multivariate

integration-by-parts argument to move the derivative away from ν to produce w. The representation in (2) is then extended to non-smooth ν.

Peng et al. [24] extend the original method of Peng et al. [23] to allow the input variables to be uniformly distributed, which is obviously an important case for many applications. Peng et al. [25] and [26] develop variance reduction techniques for GLR.

The input variable x in (2) is generally vector-valued, as is the transformation g. The range of g may have lower dimension than x, in which case GLR requires choosing an invertible submatrix of the Jacobian of g with respect to x; different choices lead to different estimators. Section 4 develops an example in which averaging over multiple such choices yields an estimator with a well-defined continuous-time limit, addressing the singularity issue raised at the end of Section 2.6. The example also suggests a connection between the GLR method and estimators derived using Malliavin calculus, as in Fournié et al. [5].

3 A Barrier Option Example

To lay groundwork for the continuous-time limit in Section 4, this section considers an option pricing example, exploring connections between GLR and combined IPA-LRM estimators.

3.1 The Option Pricing Setting

Pierre has made several contributions to simulation methodology for option pricing, including in Avramidis and L'Ecuyer [1], L'Ecuyer [19], and Lemieux and L'Ecuyer [22]. We use this setting to examine some combined derivative estimators.

A standard model in option pricing takes the underlying asset to be described by a geometric Brownian motion. We simulate the underlying asset on a discrete-time grid $0, \Delta, 2\Delta, \ldots$ according to the recursion

$$S_{i+1} = S_i \exp(\mu\Delta + \sigma\sqrt{\Delta}Z_{i+1}), \quad i = 1, 2, \ldots, \tag{3}$$

where $Z_1, Z_2, \ldots$ are independent standard normal random variables, and $S_0 = s_0 > 0$ is fixed, as are the parameters μ and $\sigma > 0$.

The payoff of the option takes the form $V(S_1, \ldots, S_n)$ or simply $V(S_n)$, with expectation $v(s_0) = \mathbb{E}[V]$. For hedging purposes, one is often interested in $v'(s_0)$. An IPA estimator differentiates V with respect to the underlying S_i and differentiates the S_i with respect to s_0 through (3). If V has discontinuities, we may apply an LRM estimator by noting that

$$\log S_1 \sim N(\log s_0 + \mu\Delta, \sigma^2\Delta), \tag{4}$$

and, given S_1, the distribution of $S_2, \ldots, S_n$ does not depend on s_0. Thus, only the distribution of S_1 contributes to the score function. With s_0 playing the role of the parameter θ, the LRM estimator takes the form (as in Broadie and Glasserman [2])

$$s(S_1)V(S_1, \ldots, S_n) = \frac{Z_1}{\sigma\sqrt{\Delta}} V(S_1, \ldots, S_n).$$

The precise form of the score function will become clear using (11), below. The main point is that we moved the dependence on s_0 from the path (through (3)) into the distribution of S_1 to be able to handle discontinuous payoffs.

3.2 The Barrier Option

One of the first examples of the GLR method in Peng et al. [23] is a barrier option. The contract has the payoff $(S_n - K)^+$ of a standard call option with strike price K, provided the underlying asset never breaches a barrier at $H > K$. The event that the path of the underlying asset terminates above K without crossing H is given by

$$A_n = \{S_i < H, i = 1, \ldots, n-1\} \cup \{K < S_n < H\}, \tag{5}$$

so we can write the payoff as

$$V = \mathbf{1}_{A_n} \cdot (S_n - K). \tag{6}$$

Write $v(H) = \mathbb{E}[V]$ for the expected payoff, and consider the problem of estimating $v'(H)$, the derivative with respect to the barrier level H. The payoff in (6) poses a challenge for derivative estimation: the payoff is piecewise constant in H, making IPA uninformative, and the distribution of $S_1, \ldots, S_n$ has no dependence on H, making LRM inapplicable.

3.3 A Combined IPA-LRM Estimator of Wang et al. [30]

To address these difficulties, we seek a change of variables that either smooths the functional dependence on H or moves H into the measure, much as we did with s_0 in (4). Set $R_i = \log(S_i/H)$, so that

$$R_1 = \log(S_0/H) + \mu\Delta + \sigma\sqrt{\Delta}Z_1, \;\; R_{i+1} = R_i + \mu\Delta + \sigma\sqrt{\Delta}Z_{i+1}, i = 1, \ldots, n-1. \tag{7}$$

Then the payoff V in (6) becomes

$$V = V(H) = \mathbf{1}\{\max_{i=1,\ldots,n} R_i < 0\}(He^{R_n} - K)^+. \tag{8}$$

This representation has a continuous functional dependence on H and a distributional dependence on H through the distribution of R_1; the conditional distribution of $R_2, \ldots, R_n$ given R_1 does not depend on H. Using IPA for the functional dependence and LRM for the distributional dependence leads to an estimator of the form

$$s_H(R_1)V + \frac{\partial V}{\partial H}, \tag{9}$$

where $s_H(R_1)$ is the score function of R_1 for parameter H.

To derive the score function for

$$R_1 \sim N(\log(S_0/H) + \mu\Delta, \sigma^2\Delta), \tag{10}$$

consider the general case

$$X \sim N(a(\theta), b^2(\theta)),$$

of a normal distribution depending on a parameter θ. Differentiating the log density with respect to θ yields the score function

$$s_\theta(X) = \left(\frac{X - a(\theta)}{b^2(\theta)}\right)\left[a'(\theta) + \frac{b'(\theta)}{b(\theta)}(X - a(\theta))\right] - \frac{b'(\theta)}{b(\theta)}. \tag{11}$$

In (10), we have $\theta = H$, $a(H) = \log(S_0/H) + \mu\Delta$, and $b \equiv \sigma\sqrt{\Delta}$. Making these substitutions, we get

$$s_H(R_1) = \frac{-Z_1}{H\sigma\sqrt{\Delta}}. \tag{12}$$

This expression for $s_H(R_1)$ uses the fact that we can recover Z_1 from R_1.

The IPA term in (9) captures the functional dependence on H in (8) and equals

$$V'(H) = \mathbf{1}_{A_n} \cdot e^{R_n}.$$

Making this substitution in (9) and transforming back from $R_1, \ldots, R_n$ to $S_1, \ldots, S_n$, we get the combined estimator

$$\mathbf{1}_{A_n} \cdot \left[(S_n - K)\frac{-Z_1}{H\sigma\sqrt{\Delta}} + \frac{S_n}{H}\right]. \tag{13}$$

This is the SLRIPA estimator of Wang et al [30].

3.4 GLR as a Combined IPA-LRM Estimator

Peng et al. [23] use the machinery of the generalized likelihood ratio method to derive an alternative estimator. The final estimator is several lines long and difficult to interpret. The estimator in (13) follows a simpler derivation and is comparatively easier to interpret because the score function and the IPA term are recognizable in the final expression. We seek a similar derivation and interpretation of the GLR estimator.

Let $R_1, \ldots, R_{n-1}$ be as before, and redefine

$$R_n = \frac{R_{n-1} + \mu\Delta + \sigma\sqrt{\Delta}Z_n + \log(H/K)}{\log(H/K)} = \frac{\log(S_n/K)}{\log(H/K)}; \tag{14}$$

this transformation is used as part of the GLR derivation in Peng et al. [23]. The payoff (6) becomes

$$V = \mathbf{1}\{\max_{i=1,\ldots,n-1} R_i < 0\}\mathbf{1}\{0 < R_n < 1\}(e^{R_n \log(H/K)} - 1)K. \tag{15}$$

As before, we will apply a combination of IPA and LRM of the form

$$s_H(R_1, \ldots, R_n)V + \frac{\partial V}{\partial H}. \tag{16}$$

To derive the relevant score function, note that R_1 still has the distribution in (10), the distribution of $R_2, \ldots, R_{n-1}$ given R_1 has no dependence on H, and

$$R_n | R_1, \ldots, R_{n-1} \sim N\left(\frac{R_{n-1} + \mu\Delta}{\log(H/K)} + 1, \frac{\sigma^2\Delta}{\log^2(H/K)}\right) \equiv N(a(H), b^2(H)). \tag{17}$$

The score function is therefore the sum of the marginal score we derived previously for R_1 in (12) and the score for the conditional density of R_n. To apply the general expression in (11)–(17), we note that

$$a'(H) = \frac{1 - a(H)}{H\log(H/K)}, \quad \frac{b'(H)}{b(H)} = -\frac{1}{H\log(H/K)}.$$

The term in square brackets in (11) is therefore

$$\frac{1 - a(H)}{H\log(H/K)} - \frac{1}{H\log(H/K)}(R_n - a(H)) = \frac{1 - R_n}{H\log(H/K)}.$$

Using (14), the factor multiplying the square brackets in (11) is

$$\frac{R_n - a(H)}{b^2(H)} = \frac{Z_n \log(H/K)}{\sigma\sqrt{\Delta}}.$$

Making these substitutions in (11) yields the score function for the conditional density of R_n. Adding the score function in (12) for R_1 yields the combined score

$$s_H(R_1, \ldots, R_n) = -\frac{Z_1}{H\sigma\sqrt{\Delta}} + \frac{Z_n}{H\sigma\sqrt{\Delta}}(1 - R_n) + \frac{1}{H\log(H/K)}. \tag{18}$$

The IPA term in (16) is given by

$$\begin{aligned} V'(H) &= \mathbf{1}\{\max_{i=1,\ldots,n-1} R_i < 0\}\mathbf{1}\{0 < R_n < 1\}e^{R_n \log(H/K)}\frac{R_n}{H}K \\ &= \mathbf{1}_{A_n}\frac{S_n \log(S_n/K)}{H\log(H/K)}. \end{aligned} \tag{19}$$

The final estimator combines (18) and (19) as in (16). A comparison with Peng et al. [23] shows that the resulting expression (16) matches the GLR estimator derived there.

This derivation shows that, in this example, the GLR estimator can be derived through a simple combination of IPA and LRM. The only difference between the derivation here and that in Sect. 3.3 is the choice of transformation R_n; the steps are otherwise the same. A strength of GLR is its generality; a benefit of the IPA-LRM derivation is that it makes the final estimator easier to interpret through the representation in (16).

4 Approaching Continuous Time: Averaging Low-Rank GLR Estimators

Consider now a more general scalar diffusion process X_t, $t \in [0, T]$, satisfying

$$dX_t = \mu(X_t)\,dt + \sigma(X_t)\,dW_t, \quad X_0 = x_0, \tag{20}$$

where W is a standard Brownian motion, x_0 is fixed, and $\mu(\cdot)$ and $\sigma(\cdot)$ are differentiable. We fix a time step $\Delta = T/n$ and simulate the discretized process

$$X_{i+1} = X_i + \mu(X_i)\Delta + \sigma(X_i)\sqrt{\Delta}Z_{i+1}, \tag{21}$$

where the Z_i are independent $N(0, 1)$ random variables. To simplify notation, we use X for both the continuous-time process in (20) and its discrete-time approximation in (21).

4.1 Approximating Continuous-Time Sensitivities

Consider an expectation of the form $v(x_0) = \mathbb{E}[V(X_n)]$, which we think of as an approximation to $\mathbb{E}[V(X_T)]$. We are interested in the sensitivity $v'(x_0)$ to the initial condition. If it is well-defined, the IPA estimator takes the form

$$V'(X_n)\frac{dX_n}{dx_0}. \tag{22}$$

Setting

$$Y_i = \frac{dX_i}{dx_0},$$

and differentiating (21) we find that these state derivatives satisfy

$$Y_{i+1} = Y_i + \mu'(X_i)Y_i\Delta + \sigma'(X_i)Y_i\sqrt{\Delta}Z_{i+1}, \tag{23}$$

with $Y_0 = 1$.

If V has discontinuities, the IPA estimator is typically biased. We can derive an LRM estimator by factoring the joint density of $X_1, \ldots, X_n$ as $f(x_0, x_1)\cdots f(x_{n-1}, x_n)$, where f is the transition density for the Markov chain (21). Only the first factor depends on x_0. Moreover, $X_1 \sim N(x_0 + \mu(x_0)\Delta, \sigma^2(x_0)\Delta)$. We thus arrive at the LRM estimator

$$s_{x_0}(X_1, \ldots, X_n)V(X_n) = \frac{Z_1}{\sigma(x_0)\sqrt{\Delta}}V(X_n). \tag{24}$$

This estimator solves the problem of discontinuous V, but it is clearly badly behaved as $\Delta \to 0$ and does not have a meaningful continuous-time limit. This makes precise the point we introduced at the end of Sect. 2.6.

Working directly in continuous time, and using tools from Malliavin calculus, Fournié et al. [5] derived the estimator

$$V(X_T)\cdot\frac{1}{T}\int_0^T \frac{Y_t}{\sigma(X_t)}\,dW_t, \tag{25}$$

where $Y_t = dX_t/dx_0$ satisfies

$$dY_t = \mu'(X_t)Y_t\,dt + \sigma'(X_t)Y_t\,dW_t, \quad Y_0 = 1.$$

The estimator in (25) is reminiscent of an LRM or GLR estimator, in the sense that it multiplies the payoff by a stochastic weight, but the connection (if any) with (24) is not immediately clear.

Chen and Glasserman [4] show that (25) can be derived as the result of averaging combinations of IPA and LRM estimators. At one extreme, (22) treats x_0 purely as

a parameter of the path of the X_i. At the other extreme, (24) treats x_0 purely as a parameter of the density of X_1. But we can consider an intermediate estimator that treats x_0 as a path parameter for $X_1, \ldots, X_{j-1}$ and then considers the density of X_n given X_{j-1}. After dropping higher-order terms, this combination results in the estimator

$$V(X_n) \cdot Y_{j-1} \frac{Z_j}{\sigma(X_{j-1})\sqrt{\Delta}}.$$

Averaging over all $j = 1, \ldots, n$ such IPA-LRM combinations yields (25) in the limit as $\Delta \to 0$.

4.2 Averaging GLR Estimators

Here we take a different approach, based on the GLR method. It is natural to look for a connection between GLR and Malliavin estimators: both use integration-by-parts formulas based on divergence operators. The Malliavin calculus versions of these terms are intended as generalizations of the classical counterparts used in GLR. We also note that Gobet and Munos [14] show that averaging derivative estimators is useful in the continuous-time context.

To put our setting in the notation of Peng et al. [23], we can write

$$X_n = g(x_0, Z_1, \ldots, Z_n),$$

with g implicitly defined through the recursion (21). The GLR estimator uses an invertible submatrix of the Jacobian (with respect to the stochastic inputs) of this transformation, and this produces an indeterminacy in the estimator when the Jacobian does not have full rank. In our setting, X_n is scalar so the Jacobian is the gradient, and the invertible submatrix reduces to the scalar

$$\bar{J}_g = \frac{\partial g}{\partial z_j}, \tag{26}$$

for any $j = 1, \ldots, n$. (Here and below, $z_1, \ldots, z_n$ are dummy variables, and $Z_1, \ldots, Z_n$ are random variables at which they are evaluated.)

Theorem 2 of Peng et al. [23] gives, under appropriate conditions,

$$\partial_{x_0} \mathbb{E}[V(X_n)] = \mathbb{E}[V(X_n) d_j(x_0, Z_1, \ldots, Z_n)],$$

where, writing φ for the standard normal density

$$d_j(x_0, z_1, \ldots, z_n) = -\text{div}\left(\partial_{x_0} g \cdot \bar{J}_g^{-1} \varphi(z_j)\right)/\varphi(z_j) = -\partial_{z_j}\left(\partial_{x_0} g \cdot \bar{J}_g^{-1} \varphi(z_j)\right)/\varphi(z_j). \tag{27}$$

In our scalar setting, the divergence operator reduces to a partial derivative with respect to z_j.

To evaluate d_j, we note that $\partial_{x_0} g = \partial_{x_0} X_n = Y_n$, and we use (21) to write

$$\bar{J}_g = \frac{\partial X_n}{\partial z_j} = \frac{\partial X_n}{\partial X_j}\frac{\partial X_j}{\partial z_j} = \frac{\partial X_n}{\partial X_j}\sigma(X_{j-1})\sqrt{\Delta}.$$

The multiplicative form of the Y_j recursion (23) yields

$$\frac{\partial X_n}{\partial X_j} = \frac{\partial X_n}{\partial X_{n-1}}\frac{\partial X_{n-1}}{\partial X_j} = \frac{Y_n}{Y_{n-1}}\frac{\partial X_{n-1}}{\partial X_j} = \frac{Y_n}{Y_{n-1}}\frac{Y_{n-1}}{Y_{n-2}}\frac{\partial X_{n-2}}{\partial X_j} = \cdots = \frac{Y_n}{Y_j}.$$

Making these substitutions, we get

$$\partial_{x_0} g \cdot \bar{J}_g^{-1} = Y_n \left(\frac{Y_n}{Y_j}\sigma(X_{j-1})\sqrt{\Delta}\right)^{-1} = \frac{Y_j}{\sigma(X_{j-1})\sqrt{\Delta}}.$$

Then, using $\varphi'(z) = -z\varphi(z)$,

$$d_j = -\partial_{z_j}\left(\frac{Y_j\varphi(Z_j)}{\sigma(X_{j-1})\sqrt{\Delta}}\right)/\varphi(Z_j) = \frac{Y_j Z_j - \sigma'(X_{j-1})Y_{j-1}\sqrt{\Delta}}{\sigma(X_{j-1})\sqrt{\Delta}}, \tag{28}$$

where we used (23) to get

$$\partial_{z_j} Y_j = \sigma'(X_{j-1})Y_{j-1}\sqrt{\Delta}.$$

Using (23), the first term in the numerator of (28) becomes

$$Y_j Z_j = [1 + \mu'(X_{j-1})\Delta + \sigma'(X_{j-1})\sqrt{\Delta}Z_j]Y_{j-1}Z_j.$$

Making this substitution and simplifying, (28) becomes

$$\begin{aligned} d_j &= \frac{Y_{j-1}Z_j}{\sigma(X_{j-1})\sqrt{\Delta}} + \frac{\mu'(X_{j-1})Y_{j-1}Z_j\sqrt{\Delta}}{\sigma(X_{j-1})} + \frac{\sigma'(X_{j-1})Y_{j-1}(Z_j^2-1)}{\sigma(X_{j-1})} \\ &\equiv \frac{Y_{j-1}Z_j}{\sigma(X_{j-1})\sqrt{\Delta}} + \epsilon_j(\Delta). \end{aligned} \tag{29}$$

Each such term is badly behaved for small Δ. But the choice of j was arbitrary; averaging over all $j = 1, \ldots, n$, and recalling that $\Delta = T/n$, we get

$$\frac{1}{n}\sum_{j=1}^{n}\frac{Y_j Z_j}{\sigma(X_{j-1})\sqrt{\Delta}} = \frac{1}{T}\sum_{j=1}^{n}\frac{Y_j Z_j\sqrt{\Delta}}{\sigma(X_{j-1})} \approx \frac{1}{T}\int_0^T \frac{Y_t}{\sigma(X_t)}\,dW_t. \tag{30}$$

The expression on the right is the Malliavin weight in (25).

To complete the argument, we need to justify dropping the $\epsilon_j(\Delta)$ in (29) and we need to turn the approximation in (30) into a limit. Conveniently, the average of the $\epsilon_j(\Delta)$ terms is precisely the error term in equation (19) of Theorem 3.1 of Chen and Glasserman [4]. We can therefore apply Theorem 4.6 of Chen and Glasserman [4] to conclude the following:

Theorem 1 *Under the conditions of Theorem 4.6 of [4], the averaged GLR estimator*

$$V(X_n)\frac{1}{n}\sum_{j=1}^{n} d_j(x_0, Z_1, \ldots, Z_n)$$

converges in distribution to the Malliavin estimator (25), as $n \to \infty$ with $\Delta = T/n$.

Although we have leveraged Chen and Glasserman [4], the derivation here is different: the estimator in Theorem 3.1 of Chen and Glasserman [4] is an average of combined IPA and LRM estimators; here we have arrived at (30) by averaging low-rank GLR estimators over alternative choices of "bases" for the low-rank representations.

Malliavin calculus is often described as a stochastic calculus of variations that provides a rigorous treatment of formal differentiation with respect to dW_t terms. The approach developed here is similar in spirit because it differentiates with respect to the Z_j, which are the discrete-time counterparts of the Brownian increments. (Neither IPA nor LRM differentiates with respect to the Z_j.) This suggests the possibility of a more systematic way to derive continuous-time derivative estimators as limits of averages of GLR estimators.

5 Concluding Remarks

We have reviewed combinations and extensions of IPA and LRM derivative estimators, a theme that dates to Pierre's early work in L'Ecuyer [17] and continues to be relevant to his recent work in Peng et al. [24]. Intriguing links between GLR and other methods open questions for further investigation in advancing a unified view of derivative estimation.

Acknowledgements I thank the editors for organizing this *Festschrift* and the reviewers for their helpful comments.

References

1. Avramidis, A., L'Ecuyer, P.: Efficient Monte Carlo and quasi-Monte Carlo option pricing under the variance gamma model. Manage. Sci. **52**(12), 1930–1944 (2006)
2. Broadie, M., Glasserman, P.: Estimating security price derivatives using simulation. Manage. Sci. **42**, 269–285 (1996)
3. Capriotti, L., Giles, M.B.: Algorithmic differentiation: adjoint Greeks made easy. Risk **25** (2012)
4. Chen, N., Glasserman, P.: Malliavin Greeks without Malliavin calculus. Stoch. Process. their Appl. **117**, 1689–1723 (2007)
5. Fournié, E., Lasry, J.-M., Lebuchoux, J., Lions, Touzi, N.: Applications of Malliavin calculus to Monte Carlo methods in finance. Financ. Stoch. **3** 391–412 (1999)
6. Giles, M.B.: Vibrato Monte Carlo. In: Monte Carlo and Quasi-Monte Carlo Methods 2008. Springer (2009)
7. Giles, M.B., Glasserman, P.: Smoking adjoints: fast Monte Carlo Greeks. Risk **19**, 88–92 (2006)
8. Glasserman, P.: Smoothing complements and randomized score functions. Ann. Oper. Res. **39**, 1–25 (1993)
9. Glasserman, P.: Monte Carlo Methods in Financial Engineering. Springer, New York (2004)
10. Glasserman, P., Yao, D.: Some guidelines and guarantees for common random numbers. Manage. Sci. **38**, 884–908 (1992)
11. Glynn, P.W.: Stochastic approximation for Monte Carlo optimization. In: Proceedings of the 1986 Winter Simulation Conference, pp. 356–365 (1986)
12. Glynn, P.W.: Optimization of stochastic systems via simulation. In: Proceedings of the 1989 Winter Simulation Conference 90–105 (1989)
13. Glynn, P.W., L'Ecuyer, P.: Likelihood ratio gradient estimation for stochastic recursions. Adv. Appl. Probab. **27**, 1019–1053 (1995)
14. Gobet, E., Munos, R.: Sensitivity analysis using Ito-Malliavin calculus and martingales, and applications to stochastic optimal control. SIAM J. Control. Optim. **43**, 1676–1713 (2005)
15. Griewal, A., Walther, A.: Evaluating Derivatives. SIAM, Philadelphia (2008)
16. Ho, Y.C., Cao, X.R.: Perturbation analysis and optimization of queueing networks. J. Optim. Theory Appl. **40**, 559–582 (1983)
17. L'Ecuyer, P.: A unified view of the IPA, SF, and LR gradient estimation techniques. Manage. Sci. **36**, 1364–1383 (1990)
18. L'Ecuyer, P.: On the interchange of derivative and expectation for likelihood ratio derivative estimators. Manage. Sci. **40**, 738–747 (1995)
19. L'Ecuyer, P.: Quasi-Monte Carlo methods with applications in finance. Finance Stoch. **13**(3), 307–349 (2009)
20. L'Ecuyer, P., Glynn, P.W.: Stochastic optimization by simulation: convergence proofs for the GI/G/1 queue in steady-state. Manage. Sci. **40**(11), 1562–1578 (1994)
21. L'Ecuyer, P., Perron, G.: On the convergence rates of IPA and FDC derivative estimators. Oper. Res. **42**(4), 643–656 (1994)
22. Lemieux, C., L'Ecuyer, P.: Efficiency improvement by lattice rules for pricing Asian options. In: Proceedings of the 1998 Winter Simulation Conference, pp. 579–586 (1998)
23. Peng, Y., Fu, M.C., Hu, J.Q., Heidergott, B.: A new unbiased stochastic derivative estimator for discontinuous sample performances with structural parameters. Oper. Res. **66**(2), 487–499 (2018)
24. Peng, Y., Fu, M., Hu, J., L'Ecuyer, P., Tuffin, B.: Generalized likelihood ratio method for stochastic models with uniform random numbers as inputs. Unpublished manuscript (2020)
25. Peng, Y., Fu, M., Hu, J., L'Ecuyer, P., Tuffin, B.: Variance reduction for generalized likelihood ratio method by conditional Monte Carlo and randomized quasi-Monte Carlo. Unpublished manuscript (2021)
26. Peng, Y., Fu, M., Hu, J., L'Ecuyer, P., Tuffin, B.: Variance reduction for generalized likelihood ratio method in quantile sensitivity estimation. In: Proceedings of the 2021 Winter Simulation Conference (2021)

27. Reiman, M.I., Weiss, A.: Sensitivity analysis for simulation via likelihood ratios. Oper. Res. **37**, 830–844 (1989)
28. Rubinstein, R.: Sensitivity analysis and performance extrapolation for computer simulation models. Oper. Res. **37**, 72–81 (1989)
29. Rubinstein, R.: Sensitivity analysis of discrete event systems by the "push out" method. Ann. Oper. Res. **39**, 229–250 (1992)
30. Wang, Y., Fu, M.C., Marcus, S.I.: A new stochastic derivative estimator for discontinuous functions with application to financial derivatives. Oper. Res. **60**, 447–460 (2012)
31. Zazanis, M., Suri, R.: Perturbation analysis gives strongly consistent sensitivity estimates for the M/G/1 queue. Manage. Sci. **34**, 39–64 (1988)
32. Zazanis, M., Suri, R.: Convergence rates of finite-difference sensitivity estimates for stochastic systems. Oper. Res. **41**, 694–703 (1993)

A Central Limit Theorem For Empirical Quantiles in the Markov Chain Setting

Peter W. Glynn and Shane G. Henderson

Abstract We provide a new proof of a central limit theorem for empirical quantiles in the positive-recurrent Markov process setting under conditions that are essentially tight. We also establish the validity of the method of nonoverlapping batch means with a fixed number of batches for interval estimation of the quantile. The conditions of these results are likely to be difficult to verify in practice, and so we also provide more easily verified sufficient conditions.

Keywords Quantile estimation · Harris processes · Regenerative processes · Markov chain Monte Carlo

1 Introduction

Given a real-valued random variable Y with cumulative distribution function (CDF) F, the pth quantile q (for $0 < p < 1$) is $q = F^{-1}(p) = \inf\{x : F(x) \geq p\}$. The problem of quantile estimation is, given p, to determine $q = F^{-1}(p)$.

We focus on the case where Y is a random variable associated with the steady-state regime of a Markov chain. To be more precise, let $X = (X_t : t \geq 0)$ be a positive (Harris) recurrent Markov chain on a general state space S in discrete or continuous time, and denote the stationary distribution of X by π. Let $f : S \to \mathbb{R}$ be a real-valued function defined on the state space S of X. We consider the problem of computing the pth quantile q of the random variable $Y = f(X_0)$, where X_0 has distribution π. Under mild additional conditions, the pth quantile Q_t of the empirical CDF

P. W. Glynn
Stanford University, Stanford, CA, USA
e-mail: glynn@stanford.edu

S. G. Henderson (✉)
Cornell University, Ithaca, NY, USA
e-mail: sgh9@cornell.edu

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_11

$$F(\cdot, t) = \frac{1}{t} \int_0^t \mathbb{1}(f(X_s) \leq \cdot)\, ds \tag{1}$$

converges to q almost surely. Our main result is a central limit theorem (CLT) for Q_t. We further show that this CLT can be leveraged to establish the validity of the nonoverlapping batch means procedure for reporting asymptotically valid confidence intervals for q.

A CLT for empirical quantiles can be established by appealing to regenerative theory. This is the approach taken in [15, 30, 39], for example, and indeed we use this approach in this paper. We exploit the "1-dependent regenerative property" of Harris processes to obtain our main results. In addition to smoothness conditions on the target CDF at the quantile q, our main assumption is that the second moment of the cycle lengths is finite. As we will show, one cannot expect the CLT to hold in general if this condition is relaxed.

Since the conditions of our main result are hard to verify in practice, we also provide more easily-verifiable conditions under which the required properties hold. These conditions are Lyapunov drift criteria, together with a condition that ensures that the target distribution is appropriately smooth at the quantile q.

Why are these particular results of interest to the simulation community? It is known that any discrete-event simulation that is "well-posed", in a certain precise sense, can be modelled as a positive Harris recurrent Markov chain [20]. If the state space of the simulation is continuous, as is often the case, then the analysis in this paper is relevant. To buttress this point we provide an example in Sect. 6.

Another application area where this problem is of great interest is in Markov chain Monte Carlo (MCMC); see, e.g., [18], [16, Chap. 5], [4, Chap. XIII] and especially [15]. In this setting, one is typically interested in exploring a given distribution π that is known only up to a normalizing constant. A Markov chain may be produced whose steady-state distribution is the given distribution π, and one then attempts to infer properties of the distribution π from Markov chain simulations. Unlike most work in MCMC, we neither assume nor require reversibility.

Quantile estimation has received a great deal of attention in the simulation community. In the case where the observations are i.i.d., [6] developed a number of important results including bias expansions that expand on the general theory for the i.i.d. case available in, e.g., [41, Sect. 2.3]. [31] derived large-deviations results for quantile estimators and explored the use of stratification techniques in estimating quantiles. Other papers that explore the use of variance reduction techniques in quantile estimation for i.i.d. observations include [19, 21, 28, 29, 37]. Additional work that develops sufficient conditions for quantile estimators that employ variance reduction techniques to satisfy a CLT includes [11, 38, 43].

In the case of estimating steady-state quantiles as in the present paper, work includes [1], where sufficient conditions for the validity of the method of nonoverlapping batch quantiles are presented, along with a practical algorithm for providing a confidence interval for a quantile. Additional practical algorithms may be found in [8, 10, 25]. Asymptotic results for the method of overlapping batch means are

stated in [45]. In closely related work, [35] gives sufficient conditions for the quantile estimator to satisfy a central limit theorem in the Markov-chain setting. The sufficient conditions ensure that the chain is geometrically ergodic through the use of Lyapunov drift criteria, along with additional conditions on the time-dependent distribution of the chain. In contrast, our conditions are much weaker and we do not require conditions on the time-dependent distribution. The sufficient conditions of [35] permit a comparison of the bias and mean-squared error of 3 estimators of steady-state quantiles in [36].

Perhaps the closest work to ours is [15], though that paper has a more practical focus on estimation methods while we strive for minimal conditions for the CLT. Reference [15] establishes the quantile CLT and describes how to estimate the variance constant that appears in the CLT using both batch means and regenerative methods. The central assumption there is *polynomial ergodicity* of an order strictly greater than 1, which implies that the length of regenerative cycles in Harris chains have a finite $(2 + \epsilon)$ moment for some $\epsilon > 0$ (see the proof of Theorem 5 in [15]), while we only require a finite 2nd moment. Moreover, polynomially ergodic chains are necessarily aperiodic; we do not require aperiodicity. Finally, [15] assumes independent regenerative cycles, yet some Harris chains arising in practice cannot have independent cycles [27].

In early work, [7, 40] established CLTs and laws of the iterated logarithm for empirical quantiles obtained from ϕ-mixing stochastic processes. Given that ergodic Markov chains are strong mixing [5, 13] one could apply these results to the Markov setting. However, we believe that the hypotheses of these results are difficult to verify in practice. The assumptions of [46] are more readily verified and were employed in the quantile estimation context by [1], but may require stronger conditions than does our analysis. For example, in the single-server example in Section 6 where we require a finite second moment condition, [14] instead requires a finite moment-generating function to verify a key assumption in [46]. Still, the machinery of [46] may be more directly applicable to some stochastic processes than ours, so the two approaches are complementary.

The remainder of this paper is organized as follows. Section 2 proves a CLT for empirical quantiles under very general hypotheses. The key hypothesis there is a uniform CLT for the empirical distribution function in a neighbourhood of the true quantile. Section 3 proves a uniform CLT for 1-dependent sequences. Section 4 specializes the results of the previous sections to obtain the desired quantile CLT for Harris processes in discrete or continuous time. Section 5 establishes the validity of non-overlapping batch means, partly through the development of a Bahadur-Ghosh representation of the quantile estimator, which may be of independent interest. Finally, Sect. 6 gives some sufficient conditions for the quantile CLT to hold, and presents a small example.

2 A Quantile Central Limit Theorem

Given a real-valued stochastic process $(W(t) : t \geq 0)$, let

$$F(\cdot, t) = t^{-1} \int_0^t \mathbb{1}(W(s) \leq \cdot)\, ds$$

be its empirical CDF. For a real-valued process $(W_k : k = 0, 1, \ldots)$ in discrete time, define $W(t) = W_{\lfloor t \rfloor}$ and $F(\cdot, t)$ as above.

For any fixed $x \in \mathbb{R}$, we say that $F(x, \cdot)$ satisfies a CLT if there exist constants $\sigma^2(x) > 0$, $F(x)$ such that for any $y \in \mathbb{R}$

$$\mathbb{P}\left(\frac{t^{1/2}[F(x,t) - F(x)]}{\sigma(x)} \leq y \right) - \Phi(y) \to 0$$

as $t \to \infty$, where Φ denotes the distribution function of a standard normal random variable. If this CLT holds, then the pointwise convergence is uniform in y, i.e.,

$$\sup_y \left| \mathbb{P}\left(\frac{t^{1/2}[F(x,t) - F(x)]}{\sigma(x)} \leq y \right) - \Phi(y) \right| \to 0$$

as $t \to \infty$; see, e.g., [41, p. 18].

We say that $F(\cdot, \cdot)$ satisfies a CLT uniformly in the set N if

$$\sup_{x \in N} \sup_y \left| \mathbb{P}\left(\frac{t^{1/2}[F(x,t) - F(x)]}{\sigma(x)} \leq y \right) - \Phi(y) \right| \to 0 \tag{2}$$

as $t \to \infty$.

Theorem 1 *Fix $q \in \mathbb{R}$ and suppose that $F(\cdot, \cdot)$ satisfies a CLT uniformly in an open neighborhood N of q. Suppose further that $F(\cdot)$ is differentiable at q, $F'(q) > 0$, $\sigma^2(q) > 0$ and $\sigma^2(\cdot)$ is continuous at q. Let $p = F(q)$ and let $Q_t = F^{-1}(p, t) = \inf\{x : F(x, t) \geq p\}$ be the pth quantile of $F(\cdot, t)$. Let*

$$G(y, t) = \mathbb{P}\left[\frac{\sqrt{t}(Q_t - q)}{\sigma(q)/F'(q)} \leq y \right].$$

Then $G(\cdot, t) \Rightarrow \Phi$ as $t \to \infty$.

Proof We employ a similar proof to the one for empirical quantiles in the i.i.d. case given in [41, p. 78]. Define $q_t = q + t^{-1/2}\sigma(q)y/F'(q)$. Then

$$G(y, t) = \mathbb{P}[Q_t \leq q_t] = \mathbb{P}[p \leq F(q_t, t)],$$

since for any cumulative distribution function H and arbitrary real x and $u \in (0, 1)$, $H^{-1}(u) \leq x$ if and only if $u \leq H(x)$ [41, Lemma 1.1.4(iii)]. Now,

$$G(y,t) = \mathbb{P}\left[t^{1/2}\frac{F(q_t,t)-F(q_t)}{\sigma(q_t)} \geq t^{1/2}\frac{p-F(q_t)}{\sigma(q_t)}\right]$$
$$= \mathbb{P}(U(q_t,t) \geq -y_t),$$

where

$$U(z,t) = \frac{t^{1/2}(F(z,t)-F(z))}{\sigma(z)} \quad \text{and} \quad y_t = \frac{t^{1/2}[F(q_t)-p]}{\sigma(q_t)},$$

and so

$$\Phi(y) - G(y,t) = \mathbb{P}[U(q_t,t) < -y_t] - (1-\Phi(y))$$
$$= [\mathbb{P}[U(q_t,t) < -y_t] - \Phi(-y_t)] + [\Phi(y) - \Phi(y_t)]. \quad (3)$$

We now show that the two bracketed terms in (3) converge to 0 as $t \to \infty$.

For the first term in (3), for t sufficiently large that $q_t \in N$,

$$|\mathbb{P}(U(q_t,t) < -y_t) - \Phi(-y_t)| \leq \sup_{x\in N}\sup_{-\infty<w<\infty} |\mathbb{P}(U(x,t) < w) - \Phi(w)|.$$

The uniform CLT assumption ensures that this term converges to 0 as $t \to \infty$.

To show that the second term in (3) converges to 0, it suffices to show that $y_t \to y$ as $t \to \infty$. Since F is differentiable at q,

$$F(q_t) - p = F(q_t) - F(q) = F'(q)(q_t - q) + o(q_t - q),$$

where a quantity r_t is said to be $o(h_t)$ if $r_t/h_t \to 0$ as $t \to \infty$. Thus,

$$y_t = \frac{F'(q)t^{1/2}(q_t-q)}{\sigma(q_t)} = \frac{\sigma(q)y + o(1)}{\sigma(q_t)}$$

as $t \to \infty$. Since $\sigma(q_t) \to \sigma(q) > 0$ as $t \to \infty$, $y_t \to y$ as $t \to \infty$. □

3 A Uniform CLT for 1-Dependent Sequences

The key ingredient in Theorem 1 is the uniform CLT. In this section we establish a uniform CLT for 1-dependent processes. We then apply this result to Harris processes in Sect. 4.

Let $Z(\theta) = (Z_n(\theta) : n \geq 1)$ be a stationary sequence of real-valued, 1-dependent, mean 0 random variables for each $\theta \in \Theta$. Let

$$S_n(\theta) = \sum_{i=1}^{n} Z_i(\theta).$$

It is well known that if $EZ_1^2(\theta) < \infty$ and

$$\eta^2(\theta) = EZ_1^2(\theta) + 2EZ_1(\theta)Z_2(\theta) > 0$$

then as $n \to \infty$, we have the CLT

$$\sup_{y \in \mathbb{R}} \left| \mathbb{P}\left(\frac{S_n(\theta)}{\eta(\theta)\sqrt{n}} \le y \right) - \Phi(y) \right| \to 0.$$

We seek a uniform (in θ) version of this result, which requires a linking assumption.

A1 The family of random variables $(Z_1^2(\theta) : \theta \in \Theta)$ is uniformly integrable.

Theorem 2 *Let $Z(\theta)$ and $\eta^2(\theta)$ be defined as above for all $\theta \in \Theta$. Suppose that Assumption A1 holds and $\eta^2(\theta)$ is bounded away from 0 for $\theta \in \Theta$. Then, as $n \to \infty$,*

$$\sup_{\theta \in \Theta} \sup_{y \in \mathbb{R}} \left| \mathbb{P}\left(\frac{S_n(\theta)}{\eta(\theta)\sqrt{n}} \le y \right) - \Phi(y) \right| \to 0.$$

We need some preliminary results before proving Theorem 2. First, we show that in proving a uniform CLT, we can ignore terms that are uniformly small in θ. The proof is an extension of that of the converging together lemma [41, p. 19] and omitted.

Lemma 1 *Let $U_n(\theta)$, $V_n(\theta)$, $W_n(\theta)$ and $X_n(\theta)$ be real-valued random variables for all $n \ge 1$ and all $\theta \in \Theta$. Suppose that for all $n \ge 1$ and all $\theta \in \Theta$, $X_n(\theta) = U_n(\theta)V_n(\theta) + W_n(\theta)$. Suppose that for all $\epsilon > 0$,*

$$\lim_{n \to \infty} \sup_{\theta \in \Theta} \mathbb{P}(|U_n(\theta) - 1| > \epsilon) = 0 \quad \text{and} \quad \lim_{n \to \infty} \sup_{\theta \in \Theta} \mathbb{P}(|W_n(\theta)| > \epsilon) = 0, \text{ and}$$

$$\lim_{n \to \infty} \sup_{\theta \in \Theta} \sup_{y} |\mathbb{P}(V_n(\theta) \le y) - G(y)| = 0$$

for some distribution function G. Then $(X_n(\cdot) : n \ge 1)$ satisfies

$$\lim_{n \to \infty} \sup_{\theta \in \Theta} \sup_{y} |\mathbb{P}(X_n(\theta) \le y) - G(y)| = 0.$$

Lemma 2 is a special case of Theorem 18.1 and Corollary 18.3 of [9].

Lemma 2 *Suppose that $(U_n : n \ge 1)$ is an i.i.d. sequence of r.v.'s with mean 0 and variance 1, and let $\mathcal{N}$ denote a standard normal random variable. Let $g(x, a) = x^2 I(|x| > a)$, and G_n denote the distribution function of $n^{-1/2} \sum_{i=1}^n U_i$. Then*

1. $\left| \mathbb{E}g\left(\frac{1}{\sqrt{n}} \sum_{i=1}^n U_i, a \right) - \mathbb{E}g(\mathcal{N}, a) \right| \le c\delta_n$, *and*
2. $\sup_y |G_n(y) - \Phi(y)| \le c\delta_n$,

where the constant c does not depend on a, n, or the distribution of U_1, *and*

$$\delta_n = \inf_{\epsilon \in [0,1]} \left(\epsilon + \mathbb{E}[U_1^2; U_1^2 > n\epsilon^2] \right).$$

Let $\tilde{Z}(\theta) = (\tilde{Z}_n(\theta) : n \geq 1)$ be an i.i.d. sequence of real-valued random variables with $\tilde{Z}_1(\theta)$ having the same distribution as $Z_1(\theta)$ for all $\theta \in \Theta$. Define the variance $\gamma^2(\theta) = \mathbb{E}Z_1^2(\theta)$, and for $n \geq 1$ let $\tilde{S}_n(\theta) = \sum_{i=1}^n \tilde{Z}_i(\theta)$.

Lemma 3 (Uniform integrability assuming independence) *Under the conditions of Theorem 2, as* $n \to \infty$,

$$\sup_{\theta \in \Theta} \mathbb{E}\left[\frac{\tilde{S}_n^2(\theta)}{n\gamma^2(\theta)}; \tilde{S}_n^2(\theta) > n\gamma^2(\theta) a_n \right] \to 0$$

for any sequence of positive constants $\{a_n\}$ *with the property that* $a_n \to \infty$ *as* $n \to \infty$.

Proof For $a > 0$, define $g(x, a) = x^2 I(|x| > a)$, and let $\mathcal{N}$ denote a standard normal random variable. Since $\mathbb{E}\mathcal{N}^2 < \infty$, $\mathbb{E}g(\mathcal{N}, a_n) \to 0$ as $n \to \infty$. Part 1 of Lemma 2 implies that

$$\left| \mathbb{E}g\left(\frac{\tilde{S}_n(\theta)}{\sqrt{n}\gamma(\theta)}, a_n \right) - \mathbb{E}g(\mathcal{N}, a_n) \right| \leq c\delta_n(\theta),$$

where c is a constant that does not depend on a_n, n or θ and

$$\delta_n(\theta) = \inf_{\epsilon \in [0,1]} \left(\epsilon + \mathbb{E}\left[\frac{\tilde{Z}_1^2(\theta)}{\gamma^2(\theta)}; \tilde{Z}_1^2(\theta) > n\epsilon^2\gamma^2(\theta) \right] \right). \tag{4}$$

We assumed that $\eta^2(\theta)$ is bounded away from 0, and therefore so is $\gamma^2(\theta)$, since

$$\eta^2(\theta) = \gamma^2(\theta) + 2\mathbb{E}Z_1(\theta)Z_2(\theta) \leq \gamma^2(\theta) + \mathbb{E}Z_1^2(\theta) + \mathbb{E}Z_2^2(\theta) = 3\gamma^2(\theta).$$

Let $\gamma^2 > 0$ be a lower bound on $\gamma^2(\theta)$ over $\theta \in \Theta$. It follows that the second term in the infimum in (4) is bounded above by

$$\gamma^{-2}\mathbb{E}[\tilde{Z}_1^2(\theta); \tilde{Z}_1^2(\theta) > n\epsilon^2\gamma^2] = \gamma^{-2}\mathbb{E}[Z_1^2(\theta); Z_1^2(\theta) > n\epsilon^2\gamma^2].$$

If we now choose $\epsilon = \epsilon(n)$ in such a way that $n\epsilon^2(n) \to \infty$ and $\epsilon(n) \to 0$ as $n \to \infty$, then A1 ensures that $\sup_{\theta \in \Theta} \delta_n(\theta) \to 0$ as $n \to \infty$, proving the result. □

Lemma 4 (Uniform integrability assuming 1-dependence) *Under the conditions of Theorem 2, as* $n \to \infty$,

$$\sup_{\theta \in \Theta} \mathbb{E}\left[\frac{S_n^2(\theta)}{n\eta^2(\theta)}; S_n^2(\theta) > n\eta^2(\theta)a_n\right] \to 0$$

for any sequence of positive constants $\{a_n\}$ *with the property that* $a_n \to \infty$ *as* $n \to \infty$.

Proof We can write

$$\begin{aligned} S_n(\theta) &= \sum_{i=1, i \text{ odd}}^{n} Z_i(\theta) + \sum_{i=1, i \text{ even}}^{n} Z_i(\theta) \\ &= \tilde{S}_n(\theta, 1) + \tilde{S}_n(\theta, 2) \text{ (say).} \end{aligned}$$

Let $M_n(\theta) = \max\{|\tilde{S}_n(\theta, 1)|, |\tilde{S}_n(\theta, 2)|\}$ so that $S_n^2(\theta) \le [2M_n(\theta)]^2$. Now, $|S_n(\theta)| > u$ implies that $|\tilde{S}_n(\theta, i)| > u/2$ for at least one of $i = 1, 2$, which is, in turn, equivalent to $M_n(\theta) > u/2$. Thus,

$$\begin{aligned} \mathbb{E}\left[\frac{S_n^2(\theta)}{n\eta^2(\theta)}; S_n^2(\theta) > n\eta^2(\theta)a_n\right] &\le \mathbb{E}\left[\frac{4M_n^2(\theta)}{n\eta^2(\theta)}; M_n^2(\theta) > n\eta^2(\theta)a_n/4\right] \\ &\le \sum_{i=1}^{2} \mathbb{E}\left[\frac{4\tilde{S}_n^2(\theta, i)}{n\eta^2(\theta)}; \tilde{S}_n^2(\theta, i) > n\eta^2(\theta)a_n/4\right]. \end{aligned} \tag{5}$$

We now apply Lemma 3 to each of the summands in (5) to complete the proof. We use the fact that each of the summands consists of essentially $n/2$ terms, and also that $\gamma^2(\theta)/\eta^2(\theta)$ is bounded away from 0 and bounded above. □

Proof (*of Theorem 2*) We use the "big block, little block" argument (e.g., [12, Theorem 7.3.1]) to reduce the problem for 1-dependent summands to one for independent summands. The big blocks are sums of consecutive $Z_i(\theta)$s, which are separated by small blocks of size 1 that ensure, together with 1-dependence, that the big blocks are independent. When the big blocks grow at an appropriate rate with n, the result follows. Let $m_n = \lfloor n^\alpha \rfloor$ be the size of the blocks, where $\alpha \in (0, 1)$. Let $k_n = \lfloor n/m_n \rfloor$ be the number of big blocks. For $1 \le j \le k_n$, define the jth big block to be

$$\Gamma_j(\theta, n) = \sum_{i=(j-1)m_n+1}^{jm_n-1} Z_i(\theta).$$

Then for $n \geq 1$,

$$\begin{aligned} S_n(\theta) &= \sum_{j=1}^{k_n} \Gamma_j(\theta, n) + \sum_{j=1}^{k_n} Z_{jm_n}(\theta) + \sum_{i=k_n m_n+1}^{n} Z_i(\theta) \\ &= S_n'(\theta) + S_n''(\theta) + S_n'''(\theta) \text{ say.} \end{aligned}$$

The hypothesis of 1-dependence ensures that for n sufficiently large, the $\Gamma_j(\theta, n)$s are i.i.d. (in j). Furthermore, so are the $Z_{jm_n}(\theta)$s provided that $m_n > 1$, which is again assured for n large enough. For any $\epsilon > 0$ and n sufficiently large that $m_n > 1$,

$$\mathbb{P}\left(\frac{|S_n''(\theta)|}{\eta(\theta)\sqrt{n}} > \epsilon\right) \leq \frac{\mathbb{E}S_n''(\theta)^2}{n\epsilon^2\eta^2(\theta)} \leq \frac{k_n\gamma^2(\theta)}{n\epsilon^2\eta^2}, \tag{6}$$

where $\eta^2 > 0$ is a lower bound on $\eta^2(\theta)$ over $\theta \in \Theta$. Assumption A1 implies that $\gamma^2(\theta)$ is bounded above, so that (6) converges to 0 uniformly in $\theta \in \Theta$ as $n \to \infty$. Similarly, we can show that $S_n'''(\theta)$ does not figure in the asymptotics (uniformly in $\theta \in \Theta$). So by Lemma 1 it suffices to show a uniform CLT for $n^{-1/2}S_n'(\theta)/\eta(\theta)$. Let

$$v_n^2(\theta) = \text{Var}\Gamma_j(\theta, n) = (m_n - 1)\eta^2(\theta) - 2EZ_1(\theta)Z_2(\theta)$$

be the variance of the big blocks. Applying Part 2 of Lemma 2 to a normalized version of $S_n'(\theta)$, we get

$$\sup_{y\in\mathbb{R}} \left| \mathbb{P}\left(\frac{S_n'(\theta)}{v_n(\theta)\sqrt{k_n}} \leq y\right) - \Phi(y) \right| \leq c\delta_n(\theta),$$

where c is a constant that does not depend on n or θ, and

$$\delta_n(\theta) = \inf_{\epsilon\in[0,1]} \left(\epsilon + \mathbb{E}\left[\frac{\Gamma_1^2(\theta, n)}{v_n^2(\theta)}; \Gamma_1^2(\theta, n) > k_n\epsilon^2 v_n^2(\theta) \right] \right). \tag{7}$$

Now choose $\epsilon = \epsilon(n)$ in such a way that $k_n\epsilon^2(n) \to \infty$ and $\epsilon(n) \to 0$ as $n \to \infty$. We then apply Lemma 4 to the second term in the infimum in (7), using the facts that $\Gamma_1(\theta, n) = S_{m_n}(\theta)$ and $v_n^2(\theta)/(m_n\eta^2(\theta)) \to 1$ as $n \to \infty$ uniformly in θ. We can then conclude that (7) converges to 0 uniformly in θ as $n \to \infty$.

To complete the proof, observe that

$$\frac{S_n'(\theta)}{v_n(\theta)\sqrt{k_n}} - \frac{S_n'(\theta)}{\eta(\theta)\sqrt{n}} = \frac{\beta_n(\theta)S_n'(\theta)}{v_n(\theta)\sqrt{k_n}},$$

where $\beta_n(\theta) \to 0$ as $n \to \infty$ uniformly in θ. Thus,

$$\mathbb{P}\left(\left|\frac{\beta_n(\theta)S_n'(\theta)}{v_n(\theta)\sqrt{k_n}}\right| > \epsilon\right) \leq \frac{\mathbb{E}S_n'(\theta)^2}{k_n v_n^2(\theta)}\frac{\beta_n^2(\theta)}{\epsilon^2} = \frac{\beta_n^2(\theta)}{\epsilon^2} \to 0$$

as $n \to \infty$, uniformly in θ. The result now follows from Lemma 1. □

4 A Quantile Central Limit Theorem for Harris Processes

We now specialize the preceding results to positive-recurrent Harris processes X on state space S in both discrete and continuous time. These processes possess 1-dependent structure that we exploit. Suppose that S is a complete, separable metric space equipped with Borel sigma algebra $\mathcal{S}$. We assume without further comment that if X is a continuous-time process, then it is non-explosive and strong Markov, and that its sample paths are right-continuous with left limits. (See [3, pp. 198–206, 407–410] for background.) Let $\mathbb{P}_x$ and $\mathbb{E}_x$ be the probability and expectation over path space when $X_0 = x$. We first define a Harris chain in discrete time.

Definition 1 We say that $X = (X_n : n = 0, 1, 2, \ldots)$ is a Harris chain on $(S, \mathcal{S})$ if there exists a set $C \in \mathcal{S}$, a $\gamma > 0$, a probability measure φ and an $m \geq 1$ such that

A2 $\mathbb{P}_x(X_m \in A) \geq \gamma\varphi(A)$ for all $x \in C$ and all $A \in \mathcal{S}$, and
A3 $\mathbb{P}_x(\sum_{n=0}^{\infty} I(X_n \in C) = \infty) = 1$ for all $x \in S$.

Harris processes in continuous time can be defined as follows.

Definition 2 We say that $X = (X_t : t \in [0, \infty))$ is a Harris process on $(S, \mathcal{S})$ if there exists a probability measure ν on $(S, \mathcal{S})$ such that whenever $\nu(A) > 0$, $\mathbb{P}_x(\int_{t=0}^{\infty} \mathbb{1}(X_t \in A) = \infty) = 1$ for all $x \in S$.

A Harris process X in discrete or continuous time automatically possesses a unique (up to a multiplicative constant) stationary measure π. If $\pi(S) < \infty$, then we can normalize π to a probability and we then say that X is positive Harris recurrent.

Harris processes are regenerative. For Harris chains (in discrete time), regeneration times can be defined through the famous split-chain construction; see [33] for a complete treatment. For Harris processes (in continuous time), regeneration times can be defined using the fact that Harris processes in continuous time observed at the event times of an independent homogeneous Poisson process are Harris chains (let us call the resulting chain the *sampled chain*), and then using the split-chain construction as discussed in [42]; see also [3, p. 199]. (Asmussen uses a non-standard definition of Harris recurrence in continuous time, but the basic ideas are present.) Here we sketch the key ideas behind this construction of regeneration times, as we will need the construction later.

Let $(\Lambda(i) : i \geq 0)$ be the event times in a homogeneous Poisson process that is independent of X, where $\Lambda(0) = 0$, and let $N(t) = \max\{i \geq 0 : \Lambda(i) \leq t\}$ for $t \geq 0$ be the associated counting process. Define $\tilde{X}_i = X_{\Lambda(i)}$ for $i \geq 0$. Then $\tilde{X} = (\tilde{X}_i : i \geq 0)$ is an embedded discrete-time Harris chain.

Proposition 1 *Let $(\tilde{X}_n : n = 0, 1, 2, \ldots)$ be the sampled chain as constructed above from a unit-rate Poisson process. Then we may assume that A2 holds with $m = 1$.*

Proof Sample the Harris process $X = (X_t : t \in [0, \infty))$ at the event times of a Poisson process with rate 2 that is independent of X to obtain a sampled chain $\hat{X} = (\hat{X}_n : n = 0, 1, 2, \ldots)$. The sampled chain $\hat{X}$ then satisfies A2 for some $m \geq 1$, C and $\gamma > 0$. Thus, for all $x \in C$, $P_x(\hat{X}_m \in \cdot) \geq \gamma\varphi(\cdot)$, i.e.,

$$\int_0^\infty \frac{2^m t^{m-1} e^{-2t}}{(m-1)!} P_x(X_t \in \cdot)\, dt \geq \gamma\varphi(\cdot).$$

We can find some $c > 0$ so that

$$ce^{-t} \geq \frac{2^m t^{m-1} e^{-2t}}{(m-1)!}$$

for all $t \geq 0$, and it follows that for all $x \in C$,

$$\int_0^\infty e^{-t} P_x(X_t \in \cdot)\, dt \geq \frac{\gamma}{c}\varphi(\cdot),$$

i.e., $P_x(\tilde{X}_1 \in \cdot) \geq (\gamma/c)\varphi(\cdot)$ for all $x \in C$, as required. □

Turning to the construction of regeneration times, if the chain is to be initiated with distribution φ then define $T(0) = 0$ (the "zeroth" regeneration time), set the number of complete regeneration cycles $\ell = 0$, a counter of "attempted splits" $n = 0$, the "wall clock time" $t = 0$ and generate $\tilde{X}_0$ from φ. Otherwise, set $T(-1) = 0$, set the number of complete regenerative cycles $\ell = -1$, the counter $n = 0$ and $t = 0$, and generate $\tilde{X}_0$ from the desired distribution of the process X at time 0. Next, generate $(\tilde{X}_1, \tilde{X}_2, \ldots, \tilde{X}_N)$, where $N = \inf\{j \geq 0 : \tilde{X}_j \in C\}$ is the (discrete) first hitting time of the set C. Also generate the (continuous) time process X up to time $\Lambda(N)$ from its appropriate conditional distribution. Next, independent of all else, set $n = n + 1$ and generate a Bernoulli random variable I_n, with $\mathbb{P}(I_n = 1) = \gamma$. If $I_n = 1$, then a regeneration occurs on the next (discrete-time) step, so set $\ell = \ell + 1$, set the ℓth regeneration time $T(\ell)$ equal to $\Lambda(N + 1)$, the time of the next event in the Poisson process beyond time $\Lambda(N)$, and generate $\tilde{X}_{N+1}$ according to φ. If $I_n = 0$, then generate $\tilde{X}_{N+1}$ according to $(\tilde{P}(x, \cdot) - \gamma\varphi(\cdot))/(1 - \gamma)$, where $\tilde{P}$ is the transition kernel for the sampled chain and $x = \tilde{X}_N$. Then generate the (continuous-time) intervening values $(X_s : \Lambda(N) < s < \Lambda(N + 1))$ from the appropriate conditional distribution given the endpoint values. Set the "current time" $t = \Lambda(N + 1)$ and repeat this process, thereby inductively constructing the continuous time process and its regeneration times $(T(k) : k \geq 0)$.

In the remainder of this section we exploit the fact that Harris processes are regenerative. In order to simultaneously treat Harris processes in both discrete and continuous time, in the remainder of this section we view a Harris chain $(X_n : n \geq 0)$ as a continuous-time process $(X_t : t \geq 0)$ where $X_t = X_{\lfloor t \rfloor}$. Such a process is no longer a Markov process, but it is regenerative.

For $i \geq 0$, let $\tau_i = T(i) - T(i-1)$ be the length of the ith regenerative cycle, and define the ith cycle to be $W_i = (X_{T(i-1)+s} : 0 \leq s < \tau_i, \tau_i)$. As discussed in [3], the cycles $(W_0, W_1, W_2, \ldots)$ are 1-dependent and the cycles $(W_1, W_2, \ldots)$ are identically distributed. This structure allows us to define the stationary measure π as follows.

For a function $g : S \to [0, \infty)$ define, for $i \geq 0$, $Y_i(g) = \int_{T(i-1)}^{T(i)} g(X_s)\, ds$. Define $Y_i(g)$ for signed g by splitting g into its positive and negative components. Now, for $A \in \mathcal{S}$, define $\pi(A) = \mathbb{E}[Y_1(\mathbb{1}(\cdot \in A))]$. Then $\pi(S) = \mathbb{E}\tau_1$, so that π has finite total mass and the process is positive Harris recurrent if and only if $\mathbb{E}\tau_1 < \infty$. We now restrict our attention to the positive Harris recurrent case, and normalize π to a probability measure by redefining $\pi(A) = \mathbb{E}[Y_1(\mathbb{1}(\cdot \in A))]/\mathbb{E}\tau_1$. Also, for $g : S \to \mathbb{R}$, define $\pi(g) = \int_S g(x)\pi(dx)$.

Now, let $f : S \to \mathbb{R}$ and for real x and $t > 0$, let $F(x, t) = t^{-1} \int_0^t \mathbb{1}(f(X_s) \leq x)\, ds$ be the empirical distribution function at time t. The strong law for positive Harris recurrent processes (see, e.g., [3, p. 203]), asserts that $F(x, t) \to F(x)$ as $t \to \infty$ almost surely, where

$$F(x) = \pi(\mathbb{1}(f(\cdot) \leq x)) = \frac{\mathbb{E}\int_{T(0)}^{T(1)} \mathbb{1}(f(X_s) \leq x)\, ds}{\mathbb{E}\tau_1}.$$

Also, let Q_t be the pth quantile associated with $F(\cdot, t)$ and q be the pth quantile of F. Our goal in this section is a CLT for Q_t.

For $t \geq 0$, let $\ell(t) = \max\{k : T(k) \leq t\}$ be the number of identically-distributed cycles completed by time t and let $\lambda = 1/\mathbb{E}\tau_1$. Also, for $i \geq 1$, define the cycle quantity

$$Z_i(x) = \int_{T(i-1)}^{T(i)} [\mathbb{1}(f(X_s) \leq x) - F(x)]\, ds.$$

Lemma 5 *Suppose that* $\mathbb{E}\tau_1^2 < \infty$. *Then*

$$\sqrt{t}(F(x, t) - F(x)) = \frac{1}{\sqrt{t}} \sum_{i=1}^{\lfloor \lambda t \rfloor} Z_i(x) + R(x, t)$$

where $\lim_{t\to\infty} \sup_x \mathbb{P}(|R(x, t)| > \epsilon) = 0$ *for any* $\epsilon > 0$.

Proof Observe that

$$t(F(x,t) - F(x)) = \int_0^t [\mathbb{1}(f(X_s) \le x) - F(x)]\, ds$$
$$= O(\tau_0) + \int_{T(0)}^{T(\ell(t))} [\mathbb{1}(f(X_s) \le x) - F(x)]\, ds + O(\tau_{\ell(t)+1}) \tag{8}$$

where $O(x)$ denotes a value z such that $|z| \le cx$ for some constant $c > 0$. We can then write (8) as

$$\sum_{i=1}^{\ell(t)} Z_i(x) + O(\tau_0 + \tau_{\ell(t)+1}). \tag{9}$$

Now, $(Z_i(x) : i \ge 1)$ is a 1-dependent, identically distributed sequence of random variables, and

$$\begin{aligned} \mathrm{Var} Z_1(x) &= \mathbb{E} Z_1^2(x) \\ &= \mathbb{E}\left(\int_{T(0)}^{T(1)} [\mathbb{1}(f(X_s) \le x) - F(x)]\, ds \right)^2 \\ &\le \mathbb{E}\left(\int_{T(0)}^{T(1)} |\mathbb{1}(f(X_s) \le x) - F(x)|\, ds \right)^2 \\ &\le \mathbb{E}\tau_1^2. \end{aligned} \tag{10}$$

From (9) we see that

$$R(x,t) = t^{-1/2} O(\tau_0 + \tau_{\ell(t)+1}) + t^{-1/2} \sum_{i=1}^{\ell(t)} Z_i(x) - t^{-1/2} \sum_{i=1}^{\lfloor \lambda t \rfloor} Z_i(x). \tag{11}$$

The first term on the right-hand side of (11) does not depend on x, and furthermore, converges almost surely to 0 as $n \to \infty$; see, e.g., [33, p. 420]. So it suffices to study the second and third terms on the right-hand side of (11). We use a modification of a standard argument (see, e.g., [33, p. 420] or [12, p. 216] for the standard case) that accounts for the 1-dependence of the sequence $(Z_i(x) : i \ge 1)$ and our goal of uniformity in x. Let ϵ and δ be arbitrary positive quantities. Then

$$\mathbb{P}\left(\left|\sum_{i=1}^{\ell(t)} Z_i(x) - \sum_{i=1}^{\lfloor \lambda t \rfloor} Z_i(x)\right| > \epsilon t^{1/2}\right)$$
$$\leq \mathbb{P}\left(\left|\sum_{i=1}^{\ell(t)} Z_i(x) - \sum_{i=1}^{\lfloor \lambda t \rfloor} Z_i(x)\right| > \epsilon t^{1/2}; |\ell(t) - \lambda t| > \delta t\right)$$
$$+ \mathbb{P}\left(\left|\sum_{i=1}^{\ell(t)} Z_i(x) - \sum_{i=1}^{\lfloor \lambda t \rfloor} Z_i(x)\right| > \epsilon t^{1/2}; |\ell(t) - \lambda t| \leq \delta t\right)$$

$$\leq \mathbb{P}(|\ell(t) - \lambda t| > \delta t) + \mathbb{P}\left(\max_{k=1}^{\lceil \delta t \rceil} \left|\sum_{i=1}^{k} Z_i(x)\right| > \epsilon t^{1/2}\right)$$
$$\leq \mathbb{P}(|\ell(t) - \lambda t| > \delta t) + \mathbb{P}\left(\max_{k=1}^{\lceil \delta t \rceil} \left|\sum_{i=1, i \text{ odd}}^{k} Z_i(x)\right| > \epsilon t^{1/2}/2\right)$$
$$+ \mathbb{P}\left(\max_{k=1}^{\lceil \delta t \rceil} \left|\sum_{i=1, i \text{ even}}^{k} Z_i(x)\right| > \epsilon t^{1/2}/2\right) \tag{12}$$
$$\leq \mathbb{P}(|\ell(t) - \lambda t| > \delta t) + \frac{(\delta t + 1)\mathrm{Var} Z_i(x)}{\epsilon^2 t/4} \tag{13}$$

where (13) follows from Kolmogorov's maximum inequality; see, e.g., [12, Theorem 5.3.1]. We add only over odd cycles or even cycles, so the terms in the sums in (12) are independent, and there are a total of at most $\delta t + 1$ terms. Now choose $\delta = \epsilon^3$, so that the bound (13) becomes

$$\mathbb{P}(|\ell(t) - \lambda t| > \delta t) + 4(\epsilon + \epsilon^{-2} t^{-1})\mathrm{Var} Z_i(x) \leq \mathbb{P}(|\ell(t) - \lambda t| > \delta t) + 4(\epsilon + \epsilon^{-2} t^{-1})\mathbb{E}\tau_1^2.$$

This bound does not depend on x, and a standard renewal-theoretic result ensures that $\ell(t)/t \to \lambda$ as $t \to \infty$ almost surely and hence in probability. Since $\epsilon > 0$ was arbitrary, this completes the proof. □

The representation given in Lemma 5 is sufficient to obtain a CLT for $F(x, \cdot)$ for any fixed x. In particular, using a CLT for 1-dependent sequences and assuming that $\mathbb{E}\tau_1^2 < \infty$ we see that $\sqrt{t}(F(x, t) - F(x)) \Rightarrow \sigma(x)N(0, 1)$ as $t \to \infty$, where

$$\sigma^2(x) = \frac{\mathbb{E}Z_1^2(x) + 2\mathbb{E}Z_1(x)Z_2(x)}{\mathbb{E}\tau_1}.$$

To apply Theorem 1 we need $\sigma^2(\cdot)$ to be continuous in a neighborhood of q. To this end we have the following result.

Lemma 6 *Suppose that $\mathbb{E}\tau_1^2 < \infty$ and that F is continuous at q. Then $\sigma^2(\cdot)$ as defined above is continuous at q.*

Proof Since $F(\cdot) = \mathbb{P}_\pi(f(X_0) \le \cdot)$ is continuous at q, it follows that $\mathbb{P}_\pi(f(X_0) = q) = 0$. But then, for all $i \ge 1$,

$$0 = \mathbb{P}_\pi(f(X_0) = q) = \frac{\mathbb{E}\int_{T(i-1)}^{T(i)} \mathbb{1}(f(X_s) = q)\, ds}{\mathbb{E}\tau_1},$$

so that $\mathbb{E}\int_{T(i-1)}^{T(i)} \mathbb{1}(f(X_s) = q)\, ds = 0$. It immediately follows that $Z_i(x) \to Z_i(q)$ as $x \to q$ almost surely for any $i \ge 1$. Hence

$$Z_1^2(x) + 2Z_1(x)Z_2(x) \to Z_1^2(q) + 2Z_1(q)Z_2(q)$$

as $x \to q$ almost surely. Furthermore, $|Z_1^2(x) + 2Z_1(x)Z_2(x)| \le \tau_1^2 + 2\tau_1\tau_2$ for any x and $\mathbb{E}(\tau_1^2 + 2\tau_1\tau_2) \le 3\mathbb{E}\tau_1^2 < \infty$. The dominated convergence theorem then gives

$$\sigma^2(x) = \frac{\mathbb{E}(Z_1^2(x) + 2Z_1(x)Z_2(x))}{\mathbb{E}\tau_1} \to \frac{\mathbb{E}(Z_1^2(q) + 2Z_1(q)Z_2(q))}{\mathbb{E}\tau_1} = \sigma^2(q)$$

as $x \to q$ as desired. □

We are now in a position to state and prove the main result of the paper.

Theorem 3 *Suppose that $\mathbb{E}\tau_1^2 < \infty$, F is differentiable at q with $F'(q) > 0$ and $\sigma^2(q) > 0$. Then, as $t \to \infty$,*

$$\frac{\sqrt{t}(Q_t - q)}{\sigma(q)/F'(q)} \Rightarrow N(0, 1).$$

Proof Lemma 6 together with the assumption that $\sigma^2(q) > 0$ ensures that $\sigma^2(\cdot)$ is bounded away from 0 in a neighborhood N of q. Furthermore, the random variables $(Z_1(x) : x \in (-\infty, \infty))$ are uniformly integrable, as can be seen from (10). These observations, together with the representation given in Lemma 5 and Theorem 2 ensure that the uniform CLT holds, i.e.,

$$\sup_{x \in N} \sup_y \left| \mathbb{P}\left(\frac{t^{1/2}[F(x,t) - F(x)]}{\sigma(x)} \le y \right) - \Phi(y) \right| \to 0$$

as $t \to \infty$. The result now follows from Theorem 1. □

Remark 1 The following example indicates that we cannot relax the assumption that $\mathbb{E}\tau_1^2 < \infty$. Let $\tau_1, \tau_2, \ldots$ be i.i.d. nonnegative random variables where $0 < \mathbb{E}\tau_1 < \infty$ and $\mathbb{E}\tau_1^2 = \infty$. For $n \ge 1$ let $T(n) = \tau_1 + \cdots + \tau_n$ and let $T(0) = 0$. For $t \ge 0$ let $\ell(t) = \sup\{n : T(n) \le t\}$ be the number of completed cycles by time t. Let $X_t = t - T(\ell(t))$, so that $X = (X_t : t \ge 0)$ is the age process associated with the renewal process $(\ell(t) : t \ge 0)$. Take f to be the identity function, and note that

$$F(x) = \frac{\mathbb{E}\int_0^{\tau_1} I(X_s \le x)}{\mathbb{E}\tau_1}\, ds = \frac{\mathbb{E}[x \wedge \tau_1]}{\mathbb{E}\tau_1},$$

where $a \wedge b = \min(a, b)$. To simplify things, we assume that $\tau_1 > 1$ a.s., so that for $x \in [0, 1]$, $F(x) = x/\mathbb{E}\tau_1$. Choose $p \in (0, 1/\mathbb{E}\tau_1)$ so that $q = F^{-1}(p) = p\mathbb{E}\tau_1 < 1$. Then for $y \in \mathbb{R}$ and t sufficiently large that $q + yt^{-1/2} < 1$,

$$\begin{aligned}
\mathbb{P}(t^{1/2}(Q_t - q) \le y) &= \mathbb{P}(Q_t \le q + yt^{-1/2}) \\
&= \mathbb{P}(p \le F(q + yt^{-1/2}, t)) \qquad (14)\\
&= \mathbb{P}(pt \le [q + yt^{-1/2}]\ell(t) + R_t), \qquad (15)
\end{aligned}$$

where $R_t = (q + yt^{-1/2}) \wedge X_t$. Equality (14) follows from [41, Lemma 1.1.4], and (15) since $tF(x, t) = x\ell(t) + x \wedge X_t$ for $x < 1$. Now, since $q = pE\tau_1$,

$$\begin{aligned}
\mathbb{P}(t^{1/2}(Q_t - q) \le y) &= \mathbb{P}\left(\frac{pt^{3/2}}{\ell(t)} - p\mathbb{E}\tau_1\sqrt{t} - \frac{R_t\sqrt{t}}{\ell(t)} \le y\right) \\
&= \mathbb{P}\left(\frac{pt}{\ell(t)}\frac{t - \ell(t)\mathbb{E}\tau_1}{\sqrt{t}} - \frac{R_t\sqrt{t}}{\ell(t)} \le y\right) \\
&= \mathbb{P}\left(\frac{pt}{\ell(t)}\frac{1}{\sqrt{t}}\sum_{i=1}^{\ell(t)}(\tau_i - \mathbb{E}\tau_1) + \frac{pt}{\ell(t)}\frac{X_t}{\sqrt{t}} - \frac{R_t\sqrt{t}}{\ell(t)} \le y\right). \qquad (16)
\end{aligned}$$

Now, $t/\ell(t) \to \mathbb{E}\tau_1$ as $t \to \infty$ a.s. Furthermore, the set of random variables $(X_t : t \ge 0)$ is tight (see [24, Proposition 1] and [23, Proposition 9]), and so

$$\frac{pt}{\ell(t)}\frac{X_t}{\sqrt{t}} - \frac{R_t\sqrt{t}}{\ell(t)} \Rightarrow 0$$

as $t \to \infty$. Thus, from the converging together lemma (e.g., [12, Theorem 4.4.6 and corollary]) and (16), $t^{1/2}(Q_t - q)$ converges in distribution to a normal random variable if and only if

$$t^{-1/2}\sum_{i=1}^{\ell(t)}(\tau_i - \mathbb{E}\tau_1) \qquad (17)$$

converges in distribution to a normal random variable. From [24], (17) converges in distribution to a normal random variable if and only if $E\tau_1^2 < \infty$, so the desired CLT does not hold. The other conditions of Theorem 3 are easily seen to hold for this example. Thus, the condition $\mathbb{E}\tau_1^2 < \infty$ is, in a certain sense, sharp.

5 The Validity of Non-overlapping Batch-Means Estimation

Theorem 3 establishes conditions under which the quantile estimator Q_t is asymptotically normally distributed. One would like to leverage this result to provide confidence intervals for q. Constructing such confidence intervals by directly estimating the variance constant $\sigma(q)/F'(q)$ is difficult, because both terms in this expression are challenging to estimate. Indeed, regenerative estimators of $\sigma(q)$ require the ability to identify the cycle boundaries $(T(i) : i \geq 0)$, and this is, at best, extremely difficult in general discrete-event simulations [27]. Furthermore, the density term $F'(q)$ requires some form of density estimator, and such estimators typically converge at a rate that is slower than the canonical $t^{-1/2}$ rate [44].

An alternative is the method of non-overlapping batch quantiles; see, e.g., [2, 35]. In this method, the sample path $(X_s : 0 \leq s \leq t)$ is divided into b batches, with the ith batch given by $(X_s : (i-1)t/b < s \leq it/b), i = 1, 2, \ldots, b$. Let $F_i(\cdot, t)$ denote the empirical CDF based on the ith batch, so that

$$F_i(x,t) = \frac{b}{t} \int_{(i-1)t/b}^{it/b} \mathbb{1}(f(X_s) \leq x)\, ds,$$

for all $x \in \mathbb{R}$ and all $i = 1, 2, \ldots, b$. Let $Q_i(t) = F_i^{-1}(p,t)$ be the estimator of the p quantile based on the ith batch. Theorem 3 basically establishes that, for each i, $Q_i(t)$ is approximately normal. If, in addition, $Q_i(t)$ is asymptotically independent of $Q_j(t)$ for $i \neq j$, then standard confidence interval theory ensures that an approximate $100(1-\alpha)\%$ confidence interval is given by

$$\bar{Q}(t) \pm t_{\alpha,b-1} \frac{s_b}{\sqrt{b}}, \tag{18}$$

where $\bar{Q}(t) = b^{-1} \sum_{i=1}^{b} Q_i(t)$ is the average of the batch quantiles, s_b^2 is the sample variance of $Q_1(t), Q_2(t), \ldots, Q_b(t)$, and $t_{\alpha,b-1}$ is the $1-\alpha/2$ quantile of a t distribution with $b-1$ degrees of freedom.

This procedure is rigorously justified through a joint CLT for $(Q_i(t) : i = 1, 2, \ldots, b)$, which we provide in Theorem 4 below.

Quantile estimators are known to exhibit bias, with the bias being on the order of the inverse of the runlength [6]. Accordingly, the estimator $\bar{Q}(t)$ has a bias that can be expected to be approximately b times as large as that of the estimator Q_t of the quantile based on the entire length-t sample path. The coverage of the confidence interval (18) can be expected to be improved if the average of the batch quantiles $\bar{Q}(t)$ is replaced by Q_t. The asymptotic validity of confidence intervals constructed in this way is assured through the joint CLT, Theorem 4, and a result that establishes that Q_t and $\bar{Q}(t)$ are "close" in the sense that $t^{1/2}(Q_t - \bar{Q}(t)) \Rightarrow 0$ as $t \to \infty$. This latter result is a direct consequence of Proposition 2 below, which gives a so-called Bahadur-Ghosh representation of quantile estimators in the Markov chain setting.

Our first result in this section provides a representation for the batch empirical CDFs along the lines of Lemma 5. The proof follows almost exactly the same lines as that of Lemma 5, using a vector version of Lemma 1, and so is omitted.

Lemma 7 *If $E\tau^2 < \infty$ then for $x \in \mathbb{R}^b$,*

$$\sqrt{\frac{t}{b}} \begin{pmatrix} F_1(x_1,t) - F(x_1) \\ F_2(x_2,t) - F(x_2) \\ \vdots \\ F_b(x_b,t) - F(x_b) \end{pmatrix} = \frac{1}{\sqrt{\frac{t}{b}}} \begin{pmatrix} \sum_{j=1}^{l} Z_j(x_1) \\ \sum_{j=l+1}^{2l} Z_j(x_2) \\ \vdots \\ \sum_{j=(b-1)l+1}^{bl} Z_j(x_2) \end{pmatrix} + R(x,t),$$

where $l = \lfloor \lambda t/b \rfloor$ and the vector-valued error term $R(x,t)$ satisfies, for any $\epsilon > 0$,

$$\lim_{t\to\infty} \sup_x \mathbb{P}(\|R(x,t)\| > \epsilon) = 0.$$

The next result is a vector version of the uniform CLT, Theorem 2. The proof is very similar to that of Theorem 2 and so we only provide a sketch of the proof.

Lemma 8 *Let $(q_1, q_2, \ldots, q_b) \in \mathbb{R}^b$ and let N_i be an open neighbourhood of q_i for each $i = 1, 2, \ldots, b$. Let $\tilde{N} = N_1 \times N_2 \times \cdots \times N_b$. If $\mathbb{E}\tau^2 < \infty$ and $\eta(x) = \mathbb{E}Z_1^2(x) + 2\mathbb{E}[Z_1(x)Z_2(x)]$ is bounded away from 0 for $x \in \cup_{i=1}^b N_i$, then*

$$\sup_{x\in\tilde{N}} \sup_{y\in\mathbb{R}^b} \left| P\left(\frac{\sum_{j=(i-1)l+1}^{il} Z_j(x_i)}{\eta(x_i)\sqrt{l}} \le y_i, i = 1, 2, \ldots, b \right) - \prod_{i=1}^{b} \Phi(y_i) \right| \to 0$$

as $t \to \infty$, where $l = l(t) = \lfloor \lambda t/b \rfloor$.

Proof (*Sketch*) Within each batch, apply the "big block little block" argument to obtain asymptotic (marginal) normality, as in the proof of Theorem 2 for each batch. To obtain the desired asymptotic independence, drop the last cycle in each batch, i.e., write the ith batch sum as

$$\sqrt{\frac{l-1}{l}} \frac{\sum_{j=(i-1)l+1}^{il-1} Z_j(x_i)}{\eta(x_i)\sqrt{l-1}} + \frac{Z_{il}(x_i)}{\eta(x_i)\sqrt{l}}$$

and now apply the matrix version of Lemma 1. □

Theorem 4 *Suppose that $\mathbb{E}\tau^2 < \infty$. Suppose further that $F(\cdot)$ is differentiable at q, $F'(q) > 0$, $\sigma^2(q) > 0$ and $\sigma^2(\cdot)$ is continuous at q. For $y \in \mathbb{R}^b$ let*

$$G(y,t) = \mathbb{P}\left(\frac{\sqrt{t/b}(Q_i(t) - q)}{\sigma(q)/F'(q)} \le y_i, i = 1, \ldots, b \right).$$

Then $G(y,t) \to \prod_{i=1}^b \Phi(y_i)$ as $t \to \infty$.

Proof The proof is very similar to that of Theorem 1. Define

$$q_{t,i} = q + \frac{\sigma(q)y_i}{F'(q)\sqrt{t/b}},$$

for $i = 1, 2, \ldots, b$. Then

$$\begin{aligned} G(y,t) &= \mathbb{P}(Q_i(t) \le q_{t,i}, i = 1, 2, \ldots, b) \\ &= \mathbb{P}(p \le F_i(q_{t,i}, t), i = 1, 2, \ldots, b) \\ &= \mathbb{P}(U_i(q_{t,i}, t) \ge -y_{t,i}, i = 1, 2, \ldots, b), \end{aligned}$$

where

$$U_i(z,t) = \sqrt{\frac{t}{b}}\frac{F_i(z,t) - F(z)}{\sigma(z)} \quad \text{and} \quad y_{t,i} = \sqrt{\frac{t}{b}}\frac{F(q_{t,i}) - p}{\sigma(q_{t,i})}.$$

Defining $\bar{\Phi}(a) = 1 - \Phi(a)$, we have that $\Phi(a) = \bar{\Phi}(-a)$, and so

$$\begin{aligned} G(y,t) - \prod_{i=1}^{b}\Phi(y_i) = \mathbb{P}(U_i(q_{t,i},t) \ge -y_{t,i}, i = 1, 2, \ldots, b) - \prod_{i=1}^{b}\bar{\Phi}(-y_{t,i}) \\ + \prod_{i=1}^{b}\Phi(y_{t,i}) - \prod_{i=1}^{b}\Phi(y_i). \end{aligned}$$

The first line of the right-hand side converges to 0 by the uniform law of large numbers. The second line converges to 0 because $y_{t,i} \to y_i$ as $t \to \infty$. □

This is the desired multivariate CLT. Thus batch means using the average of the batch quantiles is asymptotically valid.

Recall that if $t^{1/2}(Q_t - \bar{Q}(t)) \Rightarrow 0$, then we can replace the average of the batch quantiles, $\bar{Q}(t)$, in the joint CLT above with Q_t, the quantile estimator based on the entire sample path. We now establish the Bahadur-Ghosh representation

$$Q_t = q - \frac{F(q,t) - F(q)}{F'(q)} + R(t), \tag{19}$$

where $t^{1/2}R(t) \Rightarrow 0$ as $t \to \infty$. Applying this representation to each batch, $i = 1, 2, \ldots, b$ yields

$$Q_i(t) = q - \frac{F_i(q,t) - F(q)}{F'(q)} + R_i(t),$$

and averaging gives

$$\bar{Q}(t) = q - \frac{F(q,t) - F(q)}{F'(q)} + \frac{1}{b}\sum_{i=1}^{b} R_i(t)$$

$$= Q_t - R(t) + \frac{1}{b}\sum_{i=1}^{b} R_i(t)$$

which gives the desired result. It therefore remains to prove the Bahadur-Ghosh representation. We first state a lemma due to [17], and then prove the representation.

Lemma 9 *([17]) Let* $(\nu_t : t \geq 0)$ *and* $(\xi_t : t \geq 0)$ *be two stochastic processes satisfying the following conditions.*

1. *The process* $(\xi_t : t \geq 0)$ *is tight, i.e., for all* $\delta > 0$ *there exists* $M > 0$ *such that* $\mathbb{P}(|\xi_t| > M) \leq \delta$.
2. *For all* $y \in \mathbb{R}$ *and* $h > 0$,

$$\lim_{t\to\infty} \mathbb{P}(\nu_t \leq y, \xi_t \geq y + h) = \lim_{t\to\infty} \mathbb{P}(\nu_t \geq y + h, \xi_t \leq y) = 0.$$

Then $\nu_t - \xi_t \Rightarrow 0$ *as* $t \to \infty$.

Proposition 2 *Suppose that* F *is differentiable at* q *with* $F'(q) > 0$ *and* $\mathbb{E}\tau^2 < \infty$. *Then the Bahadur-Ghosh representation* (19) *is valid.*

Proof The essential elements of our proof are similar to those in [17] for the i.i.d. case. Let $y \in \mathbb{R}$ be arbitrary. As in the proof of Theorem 1, the events $\{t^{1/2}(Q_t - q) \leq y\}$ and

$$\left\{ -t^{1/2}(F(q + t^{-1/2}y, t) - F(q + t^{-1/2}y)) \leq t^{1/2}(F(q + t^{-1/2}y) - p) \right\}$$

are identical.

The differentiability of F at q ensures that $t^{1/2}(F(q + t^{-1/2}y) - p) = F'(q)y + o(1)$ as $t \to \infty$. Furthermore,

$$t^{1/2}(F(q + t^{-1/2}y, t) - F(q + t^{-1/2}y)) = t^{1/2}(F(q,t) - F(q)) + V(t),$$

where the remainder term $V(t)$ is given by

$$t^{1/2}(F(q + t^{-1/2}y, t) - F(q + t^{-1/2}y) - F(q,t) + F(q)).$$

The proof will be complete if we show that $V(t) \Rightarrow 0$ as $t \to \infty$. (To see why, take $\nu_t = t^{1/2}(Q_t - q)$ and $\xi_t = t^{1/2}(F(q,t) - F(q))/F'(q) + V(t)/F'(q)$ in Lemma 9 above.)

From Lemma 5, we can write

$$V(t) = t^{-1/2} \sum_{i=1}^{\lfloor \lambda t \rfloor} W_i(t) + R(t),$$

where the (mean-zero) cycle-term $W_i(t) = Z_i(q + t^{-1/2}y) - Z_i(q)$ and $R(t) \Rightarrow 0$ as $t \to \infty$. Chebyshev's inequality then gives that for arbitrary $\epsilon > 0$,

$$\mathbb{P}\left(\left| t^{-1/2} \sum_{i=1}^{\lfloor \lambda t \rfloor} W_i(t) \right| > \epsilon \right) \leq \frac{1}{\epsilon^2 t} \left(\lfloor \lambda t \rfloor \mathbb{E} W_1^2(t) + 2(\lfloor \lambda t \rfloor - 1) \mathbb{E}[W_1(t) W_2(t)] \right). \tag{20}$$

Now, exactly as in Lemma 6, for any fixed i, $W_i(t) \to 0$ as $t \to \infty$ a.s., and $|W_i(t)| \leq \tau_i$, and so dominated convergence ensures that the right-hand side of (20) converges to 0 as $t \to \infty$, thereby completing the proof. □

Remark 2 The Bahadur-Ghosh representation immediately provides a weak law of large numbers for the quantile estimator Q_t as well as the means to prove a CLT for Q_t based on the empirical CDF. It is natural to ask why we did not use this representation earlier in our development. An inspection of the proof of Proposition 2 shows that the essential elements of the proof are the same as those we developed in earlier sections, so it does not appear that there is anything to gain from doing so.

6 Sufficient Conditions

The assumptions of Theorems 3 and 4 are difficult to verify as stated. In this section we provide sufficient conditions for some of those assumptions that are often more easily verified in applications. Where possible, we try to give a unified treatment of both discrete-time and continuous-time Harris processes. Let $(X_t : t \geq 0)$ be a Markov process in discrete or continuous time as defined in Sect. 4. (Recall that in continuous time we assume that the process is non-explosive, strong Markov, and has sample paths that are right continuous with left limits.) We begin with the condition that the regenerative cycle lengths have finite second moment, i.e., that $E\tau_1^2 < \infty$, which can be verified through the use of drift criteria.

Definition 3 Let $X = (X_t : t \geq 0)$ be a Markov process on a complete, separable metric space in discrete or continuous time. Let $u : S \to \mathbb{R}$ and suppose that there exists $h : S \to \mathbb{R}$ such that $M = (M_t : t \geq 0)$ is a $\mathbb{P}_x$-local martingale for all $x \in S$, where

$$M_t = u(X_t) - u(X_0) - \int_0^t h(X_s)\, ds, \tag{21}$$

and t is restricted to discrete or continuous time as appropriate. We then say that u is contained in the domain $\mathcal{D}(\mathcal{A})$ of the generator $\mathcal{A}$ of X, and $\mathcal{A}u = h$.

Suppose that in addition to A2 for discrete chains, or A2 with $m = 1$ for the embedded chain for continuous-time processes, we also have the following, where the set C is as in A2 for the sampled chain.

A4 There exists $g_1 : S \to [0, \infty)$ such that for all $x \in S$, and some $b_1 > 0$,

$$\mathcal{A}g_1(x) \leq -1 + b_1 \mathbb{1}(x \in C).$$

A5 There exists $g_2 : S \to [0, \infty)$ such that for all $x \in S$ and some $b_2 > 0$,

$$\mathcal{A}g_2(x) \leq -g_1(x) + b_2 \mathbb{1}(x \in C).$$

Assumption A4 implies that X is positive-Harris recurrent; see [33, Theorem 14.0.1] for the discrete case and [34] for the continuous case. Assumptions A4 and A5 imply a finite second moment of the regeneration times, i.e., that $\mathbb{E}_\varphi \tau_1^2 < \infty$. We prove the continuous-time result; the discrete-time result follows essentially the same proof with a modest modification since m in A2 cannot be assumed to equal 1.

Lemma 10 *Suppose that A4 holds for the continuous-time process X. Let $\tilde{X}$ be the sampled chain. Then, for all x,*

$$\mathbb{E}_x g_1(\tilde{X}_1) - g_1(x) \leq -1 + b_1 \mathbb{P}_x(\tilde{X}_1 \in C). \tag{22}$$

Proof Since g_1 lies in the domain of the generator, (21) with $u = g_1$ is a $\mathbb{P}_x$ local martingale for all $x \in S$. It follows from the observations on p. 311 of [32] that

$$e^{-t} g_1(X_t) - g_1(X_0) + \int_0^t e^{-s}(g_1(X_s) - \mathcal{A}g_1(X_s))\, ds$$

is also a $\mathbb{P}_x$ local martingale for all $x \in S$. Thus, since $g_1 \geq 0$, for a sequence of stopping times $O_n \to \infty$ as $n \to \infty$ $\mathbb{P}_x$ a.s.,

$$\mathbb{E}_x[e^{-t\wedge O_n} g_1(X_{t\wedge O_n})] + \mathbb{E}_x \int_0^{t\wedge O_n} e^{-s}(g_1(X_s) - \mathcal{A}g_1(X_s))\, ds = g_1(x). \tag{23}$$

Now, A4 implies that $g_1(x) + 1 \leq g_1(x) - \mathcal{A}g_1(x) + b_1 I(x \in C)$ for all $x \in S$. Hence

$$\mathbb{E}_x \int_0^{t\wedge O_n} e^{-s}(g_1(X_s)+1)\,ds \le \mathbb{E}_x \int_0^{t\wedge O_n} e^{-s}(g_1(X_s) - \mathcal{A}g_1(X_s))\,ds + b_1\mathbb{E}_x \int_0^{t\wedge O_n} e^{-s} I(X_s \in C)\,ds \le g_1(x) + b_1\mathbb{E}_x \int_0^{\infty} e^{-s} I(X_s \in C)\,ds,$$

where, in the second inequality we used (23). Taking $n \to \infty$ and then $t \to \infty$, monotone convergence gives

$$\mathbb{E}_x \int_0^{\infty} e^{-s} g_1(X_s)\,ds + 1 \le g_1(x) + b_1\mathbb{E}_x \int_0^{\infty} e^{-s} I(X_s \in C)\,ds,$$

i.e., that $\mathbb{E}_x g_1(\tilde{X}_1) - g_1(x) \le -1 + b_1\mathbb{P}_x(\tilde{X}_1 \in C)$. □

Proposition 3 *Suppose that A4 and A5 hold. Then* $\mathbb{E}_\varphi \tau_1^2 < \infty$.

Proof Recall that we have enlarged the path space of the Markov process X to include an independent unit-rate Poisson process $(N(t) : t \ge 0)$ with event times $(\Lambda(n) : n \ge 0)$ with $\Lambda(0) = 0$ and an i.i.d. sequence of Bernoulli random variables $(I_n : n \ge 1)$ with $P(I_1 = 1) = \gamma$.

Let $\mathbb{E}_\varphi$ and $\mathbb{P}_\varphi$ denote the expectation and probability on the enlarged probability space when the chain X has initial distribution φ, so that a regeneration occurs at time 0. For convenience, write τ for τ_1. For $n \ge 0$, let $M(n) = \sum_{j=0}^{n} I(\tilde{X}_j \in C)$ be the number of attempted regenerations by time n. Define the discrete-time stopping time $\tilde{\tau} = \inf\{n \ge 0 : I_{M(n)} = 1\}$. Under $\mathbb{P}_\varphi$, the regeneration time $\tau = \Lambda(\tilde{\tau} + 1)$.

From (22) and the comparison theorem [33, Theorem 14.2.2],

$$\mathbb{E}_x \tilde{\tau} \le g_1(x) + b_1\mathbb{E}_x \sum_{j=0}^{\tilde{\tau}-1} h(\tilde{X}_j),$$

where $h(x) = \mathbb{P}_x(\tilde{X}_1 \in C)$. Since $I(\tilde{\tau} > j)$ is measurable with respect to $\mathcal{G}_j = \sigma(\tilde{X}_0, \ldots, \tilde{X}_j, I_1, \ldots, I_{M(j)})$, it follows that

$$\mathbb{E}_x \sum_{j=0}^{\tilde{\tau}-1} h(\tilde{X}_j) = \sum_{j=0}^{\infty} \mathbb{P}_x(\tilde{\tau} > j, \tilde{X}_{j+1} \in C) = \mathbb{E}_x \sum_{j=1}^{\tilde{\tau}} I(\tilde{X}_j \in C).$$

Now, each time j that $\tilde{X}_j \in C$, we regenerate with probability γ, so that $\sum_{j=1}^{\tilde{\tau}} I(\tilde{X}_j \in C)$ is geometrically distributed with success probability γ and thus has mean γ^{-1}. We conclude that $\mathbb{E}_x \tilde{\tau} \le g_1(x) + b_1/\gamma$.

With that result in hand,

$$\begin{aligned}
\mathbb{E}_\varphi \tilde{\tau}^2 &\le 2\mathbb{E}_\varphi \sum_{j=0}^{\tilde{\tau}-1} (\tilde{\tau} - j) \\
&= 2\mathbb{E}_\varphi \sum_{j=0}^{\infty} \mathbb{E}_\varphi[(\tilde{\tau} - j) I(\tilde{\tau} > j) | \mathcal{G}_j] \\
&\le 2\mathbb{E}_\varphi \sum_{j=0}^{\tilde{\tau}-1} (g_1(\tilde{X}_j) + b_1/\gamma) \\
&= 2\mathbb{E}_\varphi \sum_{j=0}^{\tilde{\tau}-1} g_1(\tilde{X}_j) + 2b_1 \mathbb{E}_\varphi \tilde{\tau}/\gamma \\
&= 2(\mathbb{E}_\varphi \tilde{\tau})\,(\mathbb{E}_\pi g_1(X_0)) + 2b_1(\mathbb{E}_\varphi \tilde{\tau})/\gamma < \infty,
\end{aligned}$$

where $\mathbb{E}_\pi g_1(X_0)$ is finite by virtue of A5 and [34, Theorem 4.2]. Since $\tau = \Lambda(\tilde{\tau} + 1)$ under $\mathbb{P}_\varphi$, Wald's second moment identity then implies that $\mathbb{E}_\varphi \tau^2 < \infty$. □

The hypotheses A4 and A5 simplify when the chain X is V-uniformly ergodic as is assumed in [35]. In fact, A4 and A5 are implied by A6 below; see, e.g., [33, Lemma 17.5.1] and [22].

A6 For the set C defined in A2 there exist constants $b, \beta > 0$, and a function $V : S \to [1, \infty)$ such that for all $x \in S$,

$$\mathcal{A}V(x) \le -\beta V(x) + bI(x \in C).$$

For the other hypotheses of Theorem 3 it is not clear exactly what form "easily verifiable" conditions should take. Indeed, it appears that one may need to tailor the conditions to a given application. It is difficult to imagine a practical application where the condition $\sigma^2(q) > 0$ would be violated, so we content ourselves with an example sufficient condition for the hypothesis that F is differentiable at q with $F'(q) > 0$. Recall that A2 and A4 imply that the chain X is positive Harris recurrent, and therefore possesses a stationary distribution, so that $F(y) = \mathbb{P}_\pi(f(X_0) \le y)$ is well-defined. In what follows we assume that X is positive Harris recurrent.

Proposition 4 *Suppose there exists a $t > 0$ such that for all y in an open neighbourhood N of q and all $x \in S$,*

$$\mathbb{P}(f(X_t) \in dy | X_0 = x) = p(x, y)dy.$$

Suppose further that for each fixed $x \in S$, $p(x, \cdot)$ is Lipschitz continuous in $y \in N$ with Lipschitz constant $L(x)$, where $L(\cdot)$ is π-integrable. Then F is differentiable

in N. If, in addition, $p(x, q) > 0$ for x in some set of positive π measure, then $F'(q) > 0$.

Proof The proof is very similar to that of Proposition 2 in [26]. Let $B = (a, b] \subseteq N$. Then

$$\begin{aligned} F(b) - F(a) &= \mathbb{P}_\pi(f(X_t) \in B) \\ &= \int_S \pi(dx)\mathbb{P}(f(X_t) \in B | X_0 = x) \\ &= \int_S \int_B \pi(dx)\, p(x, y) dy \\ &= \int_B \int_S \pi(dx)\, p(x, y) dy. \end{aligned}$$

It follows immediately that F has a density ψ in N, where

$$\psi(y) = \int_S \pi(dx)\, p(x, y) \tag{24}$$

at $y \in N$. Now, for h such that both y and $y + h \in N$,

$$|\psi(y + h) - \psi(y)| = \left| \int_S \pi(dx)(p(x, y + h) - p(x, y)) \right| \leq h \int_S \pi(dx) L(x). \tag{25}$$

Since L is π integrable, it follows that ψ is Lipschitz continuous in N. Since F has a continuous density in N, we may conclude that it is differentiable (in fact, continuously differentiable) in N with derivative ψ.

Finally, observe from (24) that the condition that $p(x, q) > 0$ for all x in a set of positive π measure implies that $\psi(q) = F'(q)$ is positive at q. □

Proposition 4 basically requires that the t-step probabilities $\mathbb{P}(f(X_t) \in dy | X_0 = x)$ have a density with respect to Lebesgue measure for all x. Typically this condition will be easiest to verify for Harris processes in discrete time in the case where $t = 1$.
Example: Consider the problem of computing quantiles of the steady-state waiting time distribution in the GI/G/1 queue. It is well-known that the sequence $X = (X_n : n \geq 0)$ of customer waiting times in the FIFO single-server queue is a Markov chain on state space $S = [0, \infty)$. In particular, X satisfies the Lindley recursion [3, p. 23] $X_{n+1} = [X_n + Y_{n+1}]^+$, where $[x]^+ = \max(x, 0)$, $Y = (Y_n : n \geq 1)$ is an i.i.d. sequence with $Y_{n+1} = V_n - U_{n+1}$, V_n is the service time of the nth customer, and U_{n+1} is the interarrival time between the nth and $(n + 1)$st customer. Take $f(x) = x$, so that we are interested in computing the quantiles of the steady-state waiting time distribution. We now verify the key conditions of Theorem 3.

As in [3, p. 23], it is straightforward to show that if $\mathbb{E}Y_1^2 < \infty$ and $\mu = \mathbb{E}Y_1 < 0$, then A4 and A5 are satisfied for the Markov chain X with $g_1(x) = 2x/|\mu|$ and $g_2(x) = 2x^2/\mu^2$. Now, for $y > 0$, we have that $P(x, dy) = \mathbb{P}(Y_1 \in d(y - x))$. So if

Y_1 has a Lipschitz continuous density with respect to Lebesgue measure and $q > 0$, then Proposition 4 implies that the distribution function F is differentiable in a neighbourhood of q. It remains to establish that $F'(q) > 0$.

First, $\pi(\{0\}) = 1 - \mathbb{E}V_1/\mathbb{E}U_1 > 0$, since $\mathbb{E}Y_1 < 0$. Furthermore, since Y_1 has a continuous density and negative mean, $\mathbb{P}(Y_1 > 0) > 0$ then implies that for each $0 \leq a < b < \infty$, there exists an $m = m(a, b)$ such that $P^m(0, (a, b)) > 0$. Therefore, $\pi((a, b)) \geq \pi(\{0\})P^m(0, (a, b)) > 0$. Proposition 4 then implies that $F'(q) > 0$.

Acknowledgements We have benefited enormously from our association with Pierre L'Ecuyer over many years. We are grateful for Pierre's scholarship, leadership and friendship. This work was partially supported by National Science Foundation grant CMMI-2035086.

References

1. Alexopoulos, C., Goldsman, D., Mokashi, A.C., Tien, K.W., Wilson, J.R.: Sequest: a sequential procedure for estimating quantiles in steady-state simulations. Oper. Res. **67**(4), 1162–1183 (2019). https://doi.org/10.1287/opre.2018.1829
2. Alexopoulos, C., Goldsman, D., Wilson, J.R.: A new perspective on batched quantile estimation. In: Laroque, C., Himmelspach, J., Pasupathy, R., Rose, O., Uhrmacher, A.M. (eds.) Proceedings of the 2012 Winter Simulation Conference. IEEE (2012)
3. Asmussen, S.: Applied Probability and Queues. Applications of Mathematics: Stochastic Modeling and Applied Probability, vol. 51, 2nd edn. Springer, New York (2003)
4. Asmussen, S., Glynn, P.W.: Stochastic Simulation: Algorithms and Analysis, Stochastic Modeling and Applied Probability, vol. 57. Springer, New York (2007)
5. Athreya, K.B., Pantula, S.G.: Mixing properties of Harris chains and autoregressive processes. J. Appl. Probab. **23**, 880–892 (1986)
6. Avramidis, A.N., Wilson, J.R.: Correlation-induction techniques for estimating quantiles in simulation experiments. Oper. Res. **46**, 574–592 (1998)
7. Babu, G.J., Singh, K.: On deviations between empirical and quantile processes for mixing random variables. J. Multivar. Anal. **8**, 532–549 (1978)
8. Bekki, J.E., Mackulak, G., Fowler, J.W., Nelson, B.L.: Indirect cycle time quantile estimation using the Cornish-Fisher expansion. IIE Trans. **42**, 31–44 (2009)
9. Bhattacharya, R.N., Rao, R.R.: Normal Approximation and Asymptotic Expansions. Wiley, New York (1976)
10. Chen, E.J., Kelton, W.D.: Quantile and tolerance-interval estimation in simulation. Eur. J. Oper. Res. **168**, 520–540 (2006)
11. Chu, F., Nakayama, M.K.: Confidence intervals for quantiles when applying variance-reduction techniques. ACM Trans. Model. Comput. Simul. **22**(2), Article 10 (2012)
12. Chung, K.L.: A Course in Probability Theory. Probability and Mathematical Statistics, vol. 21, 2nd edn. Academic, San Diego (1974)
13. Davydov, J.A.: Mixing conditions for Markov chains. Teor. Verojatnost. i Primenen. **18**, 321–338 (1973)
14. Dingeç, K.D., Alexopoulos, C., Goldsman, D., Lolos, A., Wilson, J.R.: Geometric-moment contraction of G/G/1 waiting times. In: Botev, Z. et al. (eds.) Advances in Modeling and Simulation, pp. 111–130. Springer (2022)
15. Doss, C.R., Flegal, J.M., Jones, G.L., Neath, R.C.: Markov chain Monte Carlo estimation of quantiles. Electron. J. Stat. **8**(2), 2448–2478 (2014)

16. Fishman, G.S.: Monte Carlo: Concepts, Algorithms and Applications. Springer Series in Operations Research. Springer, New York (1996)
17. Ghosh, J.K.: A new proof of the Bahadur representation of quantiles and an application. Ann. Math. Stat. **42**(6), 1957–1961 (1971)
18. Gilks, W.R., Richardson, S., Spiegelhalter, D.J. (eds.): Markov Chain Monte Carlo in Practice. Chapman & Hall, London (1996)
19. Glasserman, P., Heidelberger, P., Shahabuddin, P.: Variance reduction techniques for estimating value-at-risk. Manage. Sci. **46**, 1349–1364 (2000)
20. Glynn, P.W.: Some topics in regenerative steady-state simulation. Acta Appl. Math. **34**, 225–236 (1994)
21. Glynn, P.W.: Importance sampling for Monte Carlo estimation of quantiles. In: Proceedings of the Second International Workshop on Mathematical Methods in Stochastic Simulation and Experimental Design, pp. 180–185 (1996)
22. Glynn, P.W., Meyn, S.P.: A Liapounov bound for solutions of the Poisson equation. Ann. Probab. **24**, 916–931 (1996)
23. Glynn, P.W., Whitt, W.: Limit theorems for cumulative processes. Stoch. Process. Their Appl. **47**, 299–314 (1993)
24. Glynn, P.W., Whitt, W.: Necessary conditions in limit theorems for cumulative processes. Stoch. Process. Their Appl. **98**, 199–209 (2002)
25. Heidelberger, P., Lewis, P.A.W.: Quantile estimation in dependent sequences. Oper. Res. **32**, 185–209 (1984)
26. Henderson, S.G., Glynn, P.W.: Computing densities for Markov chains via simulation. Math. Oper. Res. **26**, 375–400 (2001)
27. Henderson, S.G., Glynn, P.W.: Regenerative steady-state simulation of discrete event systems. ACM Trans. Model. Comput. Simul. **11**, 313–345 (2001)
28. Hesterberg, T.C., Nelson, B.L.: Control variates for probability and quantile estimation. Manage. Sci. **44**(9), 1295–1312 (1998)
29. Hsu, J.C., Nelson, B.L.: Control variates for quantile estimation. Manage. Sci. **36**, 835–851 (1990)
30. Iglehart, D.L.: Simulating stable stochastic systems, VI: quantile estimation. J. Assoc. Comput. Mach. **23**, 347–360 (1976)
31. Jin, X., Fu, M.C., Xiong, X.: Probabilistic error bounds for simulation quantile estimators. Manage. Sci. **49**, 230–246 (2003)
32. Karlin, S., Taylor, H.M.: A Second Course in Stochastic Processes. Academic, Boston (1981)
33. Meyn, S.P., Tweedie, R.L.: Markov Chains and Stochastic Stability. Springer, London (1993)
34. Meyn, S.P., Tweedie, R.L.: Stability of Markovian processes III: Foster-Lyapunov criteria for continuous-time processes. Adv. Appl. Probab. **25**, 518–548 (1993)
35. Muñoz, D.F.: On the validity of the batch quantile method for Markov chains. Oper. Res. Lett. **38**, 223–226 (2010)
36. Muñoz, D.F., Ramírez-López, A.: A note on bias and mean squared error in steady-state quantile estimation. Oper. Res. Lett. **43**, 374–377 (2015)
37. Nakayama, M.K.: Asymptotically valid confidence intervals for quantiles and values-at-risk when applying Latin hypercube sampling. Int. J. Adv. Syst. Meas. **4**, 86–94 (2011)
38. Nakayama, M.K.: Using sectioning to construct confidence intervals for quantiles when applying importance sampling. In: Laroque, C., Himmelspach, J., Pasupathy, R. Rose, O., Uhrmacher, A.M. (eds.) Proceedings of the 2012 Winter Simulation Conference. IEEE (2012)
39. Seila, A.F.: A batching approach to quantile estimation in regenerative simulations. Manage. Sci. **28**, 573–581 (1982)
40. Sen, P.K.: On the Bahadur representation of sample quantiles for sequences of ϕ-mixing random variables. J. Multivar. Anal. **2**, 77–95 (1972)
41. Serfling, R.J.: Approximation Theorems of Mathematical Statistics. Wiley, New York (1980)
42. Sigman, K.: One-dependent regenerative processes and queues in continuous time. Math. Oper. Res. **15**(1), 175–189 (1990)

43. Sun, L., Hong, L.J.: Asymptotic representations for importance-sampling estimators of value-at-risk and conditional value-at-risk. Oper. Res. Lett. **38**, 246–251 (2010)
44. Wand, M., Jones, M.: Kernel Smoothing. Chapman & Hall, London (1995)
45. Wood, D.C., Schmeiser, B.W.: Overlapping batch quantiles. In: Alexopoulos, C., Kang, K., Lilegdon, W.R., Goldsman, D. (eds.) Proceedings of the 1995 Winter Simulation Conference, pp. 303–308. IEEE, Piscataway, New Jersey (1995)
46. Wu, W.B.: On the Bahadur representation of sample quantiles for dependent sequences. Ann. Stat. **33**(4), 1934–1963 (2005)

Simulation of Markov Chains with Continuous State Space by Using Simple Stratified and Sudoku Latin Square Sampling

Rami El Haddad, Joseph El Maalouf, Rana Fakhereddine, and Christian Lécot

Abstract Monte Carlo (MC) is widely used for simulating discrete time Markov chains. Here, N copies of the chain are simulated in parallel, using pseudorandom numbers. We restrict ourselves to a one-dimensional continuous state space. We analyze the effect of replacing pseudorandom numbers on $I := [0, 1)$ with stratified random points over I^2: for each point, the first component is used to select a state and the second component is used to advance the chain by one step. Two stratified sampling techniques are compared: simple stratified sampling (SSS) and Sudoku Latin square sampling (SLSS). For both methods and for $N = p^2$ samples, the unit square is dissected into p^2 subsquares and there is one sample in each subsquare. For SLSS, each side of the unit square is divided into N subintervals and the projections of the samples on the side are distributed with one projection in each subinterval. Stratified strategies outperform classical MC if the N copies are reordered by increasing states at each step. We prove that the variance of SSS and SLSS estimators is bounded by $\mathcal{O}(N^{-3/2})$, while it is bounded by $\mathcal{O}(N^{-1})$ for MC. The results of numerical experiments indicate that these upper bounds match the observed rates. They also show that SLSS gives a smaller variance than SSS.

Keywords Monte Carlo methods · Markov chains · Simulation · Stratified sampling

R. El Haddad (✉) · R. Fakhereddine
Laboratoire de Mathématiques et Applications, U.R. Mathématiques et modélisation, Faculté des sciences, Université Saint-Joseph, B.P. 7-5208, Mar Mikhaël, 1104 2020 Beirut, Lebanon
e-mail: rami.haddad@usj.edu.lb

R. Fakhereddine
e-mail: rana.fakhreddine1@usj.edu.lb

J. El Maalouf
College of Engineering and Technology, American University of the Middle East, Egaila, Kuwait
e-mail: joseph.el-maalouf@aum.edu.kw

C. Lécot
Université Savoie Mont Blanc, Laboratoire de Mathématiques, UMR 5127 CNRS, Le Bourget-du-Lac 73376, France
e-mail: christian.lecot@univ-smb.fr

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_12

1 Introduction

A number of practical systems can be modeled as Markov chains with a large state space. Fields of application are particles physics, telecommunications, queueing theory, mathematical finance, etc. In many situations, analytic solutions are not available and deterministic numerical methods are not practicable. Monte Carlo (MC) simulation has become the classical way to solve such problems.

We consider a discrete time Markov chain with a one-dimensional state space. The discrete case has been considered in a recent paper [4], and we turn now to the case of a continuous state space. We restrict ourselves to $\mathbb{R}$, for simplification purposes. We also assume that only one random variate is used to advance the chain by one step. The chain evolves according to the recurrence:

$$X_{n+1} = \varphi_{n+1}(X_n, U_{n+1}), \quad n \geq 0, \tag{1}$$

where $U_1, U_2, \ldots$ are i.i.d. uniform random variables over $I := [0, 1)$, and $\varphi_1, \varphi_2, \ldots$ are measurable functions $\mathbb{R} \times I \to \mathbb{R}$. In what follows, P_n denotes the distribution law of X_n and F_n its cumulative distribution function. MC methods use pseudorandom numbers as realizations of uniform random variables over I. The drawback is that convergence can be slow. For example, we show below that the variance of the MC estimator for any F_n behaves as $\mathcal{O}(N^{-1})$, if N copies of the chain are simulated in parallel.

A possible step towards improving the accuracy of MC methods is to apply quasi-Monte Carlo (QMC) methods [3, 20]. Let d denote a dimension and λ_d be the d-dimensional Lebesgue measure. For QMC integration over I^d, small errors are guaranteed if points with small discrepancy (quasirandom points) are used. Let $|\mathcal{E}|$ denote the cardinality of a (finite) set $\mathcal{E}$. The *star discrepancy* of a set $\mathcal{U}$ of N points in I^d is defined by

$$D_N^\star(\mathcal{U}) := \sup_{J^\star} \left| \frac{|\mathcal{U} \cap J^\star|}{N} - \lambda_d(J^\star) \right|,$$

where $J^\star$ runs through all subintervals of I^d anchored at the origin. This concept may be adapted for a probability measure P on $\mathbb{R}^d$: one defines the *P-discrepancy* of a set $\mathcal{X}$ of N points in $\mathbb{R}^d$ as follows:

$$D_N^\star(\mathcal{X}; P) := \sup_{J^\star} \left| \frac{|\mathcal{X} \cap J^\star|}{N} - P(J^\star) \right|,$$

where $J^\star$ runs through all products of left-unbounded intervals.

QMC simulation of Markov chains has been proposed and analyzed in [13]. A number $N = b^m$ of paths are simulated in parallel. One chooses N states $x_1^0, x_2^0, \ldots, x_N^0$ with a small P_0-discrepancy. To advance on the paths, a specific low discrepancy sequence, so-called $(0, 2)$-sequence in base b (see [3, 20]), is used : $u_1, u_2, \ldots$ From step n to $n + 1$, one utilizes the point u_{nN+k} $(1 \leq k \leq N)$ as follows:

the first component chooses the chain and the second component determines the new state. This procedure is efficient if the chains are reordered according to their states at each step: $x_1^n \leq x_2^n \leq \cdots \leq x_N^n$. Numerical experiments show that to replace pseudorandom numbers by quasirandom numbers, without an additional measure such as reordering, is useless (see also [19]). The accuracy of this QMC method can be estimated at step n by the P_n-discrepancy of the states: the theoretical bound in [13] is of order $\mathcal{O}(N^{-0.5})$ when N chains are simulated; various numerical experiments show that this discrepancy behaves between $\mathcal{O}(N^{-0.7})$ and $\mathcal{O}(N^{-0.8})$ (see [12]).

A *randomized quasi-Monte Carlo* approach (Array-RQMC) for simulating Markov chains has been proposed and analyzed in [15–17]. For a one-dimensional state space, with N copies of the chain, a bound for the variance of the estimator was obtained of order $\mathcal{O}(N^{-3/2})$; but one needs an independence assumption between the random numbers used, which is not always satisfied in practice (see Sect. 3 of [17]). A comparable result was obtained in [9]: in the context of particle filters, the authors proved that the Array-RQMC estimator converges as $o(N^{-1})$.

When the state space is higher-dimensional, say ℓ or when more random variates are needed to advance the chain, say d, then $(\ell + d)$-dimensional random samples are used. The first ℓ coordinates of the points match the states to the points and the last d coordinates determine the next states. In addition, a multivariate sort is employed to order the states. Array-RQMC methods in the multidimensional case are studied in [1, 15, 18, 25]. The same problem was addressed with QMC methods in [6, 7, 12].

Variance reduction through stratified sampling was studied on experiments in [5, 8] and theoretically analyzed for Markov chains with a discrete state space in [4]. In dimension two and for an integer p, we consider two kinds of stratified point sets $\{x_k : 1 \leq k \leq p^2\}$.

- *Simple stratified sampling* (SSS) satisfies: if I^2 is divided into squares J_k of area $1/p^2$, there is one x_k in each J_k.
- *Sudoku Latin square sampling* (SLSS) also satisfies: if I is divided into intervals H_k of length $1/p^2$, then, for $i = 1, 2$, there is one projection $x_{k,i}$ in each H_k. An example is shown on Fig. 1.

Simple stratified sampling has been introduced in [11] and further studied in [2]. Sudoku Latin square sampling is a special case of the *orthogonal array based Latin hypercube sampling* of [26]. The context of this stratification is more thoroughly detailed in [4, 5], where several references [21–24, 27] are discussed.

In the case of Markov chains with a one-dimensional discrete state space, we have proved a bound for the variance of order $\mathcal{O}(N^{-3/2})$ for both stratified strategies and this order is observed on numerical experiments [4]. In the present paper, we supplement this analysis by considering Markov chains with a one-dimensional continuous state space. We follow the same steps as in the discrete case, and obtain bounds of the same order, which, in discrete or continuous cases, match the rates observed in numerical experiments. One must notice that, in both cases, we bound the variance of a Bernoulli random variable (Lemma 2) or the covariances of pairs of Bernoulli

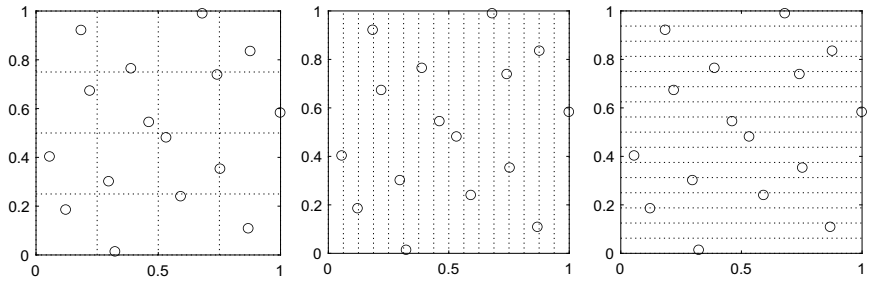


Fig. 1 A Sudoku Latin square sample of 4^2 points

random variables (Lemma 3). We prove the $\mathcal{O}(N^{-3/2})$ upper bound for the variance of the SLSS method (Lemma 3 and Proposition 3) without making the independence assumption of [17] (which is not satisfied for this scheme). One step of the proof amounts to proving a bound of the same order for the SSS method, and we give it separately (Lemma 2 and Proposition 2), for the sake of clarity, although this result is not new.

In Sect. 2, we present three stochastic methods for simulating Markov chains with a one-dimensional state space: standard MC, SSS and SLSS. In Sect. 3, we provide bounds for the variance of the estimator of the cumulative distribution function. In Sect. 4 we present the results of numerical experiments, with various Markov chains; we compute the empirical variances of the estimators and compare them with our theoretical bounds. Conclusions are drawn in Sect. 5.

2 Markov Chain Simulation with Stratified Sampling

Consider a chain with the recurrence given in (1). Denote by $\mathcal{B}_+(\mathbb{R})$ the set of nonnegative bounded measurable functions defined on $\mathbb{R}$. For every $s \in \mathcal{B}_+(\mathbb{R})$ we have

$$\int_{\mathbb{R}} s(x) dP_{n+1}(x) = \int_{\mathbb{R}\times I} s \circ \varphi_{n+1}(x,u) dP_n(x) du. \tag{2}$$

For the numerical solution, we choose an integer N. We suppose that we approximate (in a sense that will be precised later) the initial distribution P_0 by a discrete uniform probability distribution concentrated on a set of N deterministic points x_k^0 (the states),

$$\widehat{P}_0 := \frac{1}{N} \sum_{k=1}^{N} \delta_{x_k^0}.$$

Here δ_x denotes the Dirac measure at $x \in \mathbb{R}$. We advance the process: suppose that P_n is approximated by (we are still using deterministic points):

$$\widehat{P}_n := \frac{1}{N} \sum_{k=1}^{N} \delta_{x_k^n}$$

and we want to find the new states at step $n+1$. Following (2), an approximation $\widetilde{P}_{n+1}$ of P_{n+1} would satisfy, for every $s \in \mathcal{B}_+(\mathbb{R})$:

$$\int_{\mathbb{R}} s(x) d\widetilde{P}_{n+1}(x) = \frac{1}{N} \sum_{k=1}^{N} \int_I s \circ \varphi_{n+1}(x_k^n, u) du. \tag{3}$$

For $1 \le k \le N$, let 1_k denote the indicator function of $H_k := [(k-1)/N, k/N)$. We associate to any $s \in \mathcal{B}_+(\mathbb{R})$ the following function of two variables:

$$C_s^n(u) := \sum_{k=1}^{N} 1_k(u_1) s \circ \varphi_{n+1}(x_k^n, u_2), \quad u = (u_1, u_2) \in I^2.$$

For $v \in I$, denote $\kappa(v) := \lfloor Nv \rfloor + 1$, then $C_s^n(u) = s \circ \varphi_{n+1}(x_{\kappa(u_1)}^n, u_2)$. It is easily shown that

$$\int_{\mathbb{R}} s(x) d\widetilde{P}_{n+1}(x) = \int_{I^2} C_s^n(u) du. \tag{4}$$

In the following sections, we present three stochastic algorithms to simulate (1). If m is an integer, we denote $[1, m] := \{1, 2, \ldots, m\}$. The notation $U \sim \mathfrak{U}(\mathcal{E})$ means that U is a random variable uniformly distributed over the set $\mathcal{E}$. If $w \in \mathbb{R}$, then s_w denotes the indicator function of the interval $(-\infty, w)$.

2.1 Classical Monte Carlo

For initialization, let $\{X_k^0 : 1 \le k \le N\}$ be i.i.d. random variables with probability distribution P_0. Then, for any $s \in \mathcal{B}_+(\mathbb{R})$,

$$\mathbb{E}\left[\frac{1}{N} \sum_{k=1}^{N} s(X_k^0)\right] = \int_{\mathbb{R}} s(x) dP_0(x), \tag{5}$$

and there exists some $\alpha_0 > 0$, with $\alpha_0 \le 1/4$ such that, for every $w \in \mathbb{R}$,

$$\mathrm{Var}\left(\frac{1}{N} \sum_{k=1}^{N} s_w(X_k^0)\right) \le \frac{\alpha_0}{N}. \tag{6}$$

We suppose that we have generated a set of random variables $\{X_1^n, X_2^n, \ldots, X_N^n\}$, which are realizations of the chains at step n. The transition from n to $n+1$ acts as follows: Let $\{U_k^{n+1} : 1 \le k \le N\}$ be independent random variables with $U_k^{n+1} \sim \mathfrak{U}(I)$. We replace the right-hand side of (4) with the following Monte Carlo estimate:

$$\widehat{X}_s^{n+1} := \frac{1}{N}\sum_{k=1}^{N} C_s^n\left(\frac{k-1}{N}, U_k^{n+1}\right) = \frac{1}{N}\sum_{k=1}^{N} s \circ \varphi_{n+1}(X_k^n, U_k^{n+1}).$$

And we generate

$$X_k^{n+1} = \varphi_{n+1}(X_k^n, U_k^{n+1}),$$

so that

$$\widehat{X}_s^{n+1} = \int_{\mathbb{R}} s(x)d\widehat{P}_{n+1}(x).$$

2.2 Simple Stratified Sampling

We choose $N = p^2$, for some integer $p > 0$. For initialization, let $\{Y_k^0 : 1 \le k \le N\}$ be independent random variables. We assume the following.

1. For any $s \in \mathcal{B}_+(\mathbb{R})$, (5) is satisfied by the Y_k^0.
2. There exists some $\beta_0 > 0$ such that, for every $w \in \mathbb{R}$,

$$\mathrm{Var}\left(\frac{1}{N}\sum_{k=1}^{N} s_w(Y_k^0)\right) \le \frac{\beta_0}{N^{3/2}}. \tag{7}$$

This may be done by the inversion method as follows. We assume that P_0 has density f_0. Let $\{V_k^0 : 1 \le k \le N\}$ be independent random variables, with $V_k^0 \sim \mathfrak{U}(H_k)$. If $Y_k^0 := F_0^{-1}(V_k^0)$, then (5) is satisfied and

$$\mathrm{Var}\left(\frac{1}{N}\sum_{k=1}^{N} s_w(Y_k^0)\right) \le \frac{1}{4N^2}.$$

We suppose that we have generated a set of random variables $\{Y_1^n, Y_2^n, \ldots, Y_N^n\}$, which are realizations of the chains at step n. The transition from n to $n+1$ has two steps: renumbering the chains and numerical integration.

(S1) The chains are relabeled so that $Y_1^n \le Y_2^n \le \cdots \le Y_N^n$. The technique was initiated in the QMC context and used for simulation of Markov chains in [13]; it guarantees theoretical and numerical convergence of the scheme.

(S2) Consider a partition of I^2 into N squares: $J_\ell = I_{\ell_1} \times I_{\ell_2}$, where, for $\ell = (\ell_1, \ell_2) \in [1, p]^2$, $I_{\ell_1} := [(\ell_1 - 1)/p, \ell_1/p)$ and $I_{\ell_2} := [(\ell_2 - 1)/p, \ell_2/p)$. Let $\{V_\ell^{n+1} : \ell \in [1, p]^2\}$ be independent random variables, where $V_\ell^{n+1} = (V_{\ell,1}^{n+1}, V_{\ell,2}^{n+1}) \sim \mathfrak{U}(J_\ell)$. We replace the right-hand side of (4) with its simple stratified estimate:

$$\widehat{Y}_s^{n+1} := \frac{1}{N} \sum_{\ell \in [1,p]^2} C_s^n(V_\ell^{n+1}) = \frac{1}{N} \sum_{\ell \in [1,p]^2} \sum_{k=1}^{N} 1_k(V_{\ell,1}^{n+1}) s \circ \varphi_{n+1}(Y_k^n, V_{\ell,2}^{n+1}).$$

And we generate

$$Y_{(\ell_1-1)p+\ell_2}^{n+1} = \varphi_{n+1}\left(Y_{\kappa(V_{\ell,1}^{n+1})}^n V_{\ell,2}^{n+1}\right),$$

so that

$$\widehat{Y}_s^{n+1} = \int_{\mathbb{R}} s(x) d\widehat{P}_{n+1}(x).$$

The first projection $V_{\ell,1}^{n+1}$ of V_ℓ^{n+1} is used for selecting the state at step n and the second projection $V_{\ell,2}^{n+1}$ is used for advancing the chain by one step. The mapping between the p^2 points and the N states is not necessarily one-to-one: it is possible to choose the same state more than once and leave out some of them. This differs from the SSS scheme used in [17, 18].

2.3 Sudoku Latin Square Sampling

We choose again $N = p^2$, for some $p > 0$. Initialization is done as before: $\{Z_k^0 : 1 \le k \le N\}$ are independent random variables. We assume the following.

1. For any $s \in \mathcal{B}_+(\mathbb{R})$, (5) is satisfied by the Z_k^0.
2. There exists some $\gamma_0 > 0$ such that, for every $w \in \mathbb{R}$,

$$\mathrm{Var}\left(\frac{1}{N} \sum_{k=1}^{N} s_w(Z_k^0)\right) \le \frac{\gamma_0}{N^{3/2}}. \tag{8}$$

As for simple stratified sampling, this may be done by inversion. We suppose that we have generated a set of random variables $\{Z_1^n, Z_2^n, \ldots, Z_N^n\}$, which are realizations of the chains at step n. The transition from n to $n + 1$ has two steps as above.

(S1) The chains are relabeled so that $Z_1^n \le Z_2^n \le \cdots \le Z_N^n$.

(S2) We consider the same partition of I^2: J_ℓ, $\ell \in [1, p]^2$ as before. For $\ell \in [1, p]^2$, let $W_\ell^{n+1} = (W_{\ell,1}^{n+1}, W_{\ell,2}^{n+1})$ be random variables, with

$$W_{\ell,1}^{n+1} := \frac{\ell_1 - 1}{p} + \frac{\sigma_1^{n+1}(\ell_2) - 1 + U_{\ell,1}^{n+1}}{p^2} \quad W_{\ell,2}^{n+1} := \frac{\ell_2 - 1}{p} + \frac{\sigma_2^{n+1}(\ell_1) - 1 + U_{\ell,2}^{n+1}}{p^2}.$$

Here, $\sigma_i^{n+1}, i = 1, 2$, are random permutations of $[1, p]$ and $U_\ell^{n+1} = (U_{\ell,1}^{n+1}, U_{\ell,2}^{n+1}) \sim \mathfrak{U}(I^2)$; all these random variables being independent. We have $W_\ell^{n+1} \sim \mathfrak{U}(J_\ell)$; in addition, the maps $\ell \to \kappa(W_{\ell,1}^{n+1})$ and $\ell \to \kappa(W_{\ell,2}^{n+1})$ are (random) bijections from $[1, p]^2$ to $[1, N]$. We replace the right-hand side of (4) with its Sudoku Latin square estimate

$$\widehat{Z}_s^{n+1} := \frac{1}{N} \sum_{\ell \in [1,p]^2} C_s^n(W_\ell^{n+1}) = \frac{1}{N} \sum_{\ell \in [1,p]^2} \sum_{k=1}^{N} 1_k(W_{\ell,1}^{n+1}) s \circ \varphi_{n+1}(Z_k^n, W_{\ell,2}^{n+1}).$$

And we generate

$$Z_{(\ell_1-1)p+\ell_2}^{n+1} = \varphi_{n+1}\left(Z_{\kappa(W_{\ell,1}^{n+1})}^n W_{\ell,2}^{n+1}\right),$$

so that

$$\widehat{Z}_s^{n+1} = \int_{\mathbb{R}} s(x) d\widehat{P}_{n+1}(x).$$

As before, $W_{\ell,1}^{n+1}$ is used for selecting the state at step n and $W_{\ell,2}^{n+1}$ is used for advancing the chain. Now, the mapping between the p^2 points and the N states is one-to-one: each state is chosen once.

3 Variance Bounds

We consider the three stochastic schemes previously introduced. For each of them, we show that, for any $s \in \mathcal{B}_+(\mathbb{R})$ and any $n \geq 0$, the estimator of $\int_{\mathbb{R}} s(x) d P_n(x)$ is unbiased. Then, we prove that for any $w \in \mathbb{R}$ and any n, the variance of the estimator of $\int_{\mathbb{R}} s_w(x) d P_n(x)$ is bounded by $\mathcal{O}(N^{-1})$ for standard Monte Carlo and by $\mathcal{O}(N^{-3/2})$ for both stratification strategies.

3.1 Classical Monte Carlo

In the following lemma, we focus on one step (from n to $n + 1$) of the algorithm and we assume that the values $x_1^n, x_2^n, \ldots, x_N^n$ of $X_1^n, X_2^n, \ldots, X_N^n$ are given numbers. We consider the estimator $\widehat{X}_s^{n+1}$ of $\int_{\mathbb{R}} s(x) d\widetilde{P}_{n+1}(x)$.

Lemma 1 *For the classical Monte Carlo method, we have:*

1. *For any* $s \in \mathcal{B}_+(\mathbb{R})$,
$$\mathbb{E}[\widehat{X}_s^{n+1}] = \int_{\mathbb{R}} s(x) d\widetilde{P}_{n+1}(x).$$
2. *For any* $w \in \mathbb{R}$,
$$\mathrm{Var}(\widehat{X}_{s_w}^{n+1}) \leq \frac{1}{4N}.$$

Proof 1. We have
$$\mathbb{E}[\widehat{X}_s^{n+1}] = \frac{1}{N} \sum_{k=1}^{N} \int_I s \circ \varphi_{n+1}(x_k^n, u) du,$$
and the result follows from (3).

2. Each variable $s_w \circ \varphi_{n+1}(x_k^n, U_k^{n+1})$ is a Bernoulli random variable, with variance $\leq 1/4$. Hence the result.

□

We then obtain a variance bound by using techniques employed in [17]. We suppose from now on that P_0 has density f_0. In addition, we make the following assumptions.

A1. For any $x \in \mathbb{R}$, the mapping $u \in I \to \varphi_{n+1}(x, u) \in \mathbb{R}$ is strictly increasing and bijective; hence we can define a function $y \in \mathbb{R} \to \Phi_{n+1}(x, y) \in I$ which is strictly increasing and bijective, such that $y = \varphi_{n+1}(x, u) \Leftrightarrow u = \Phi_{n+1}(x, y)$.

A2. For any $x \in \mathbb{R}$, the mapping $y \to \Phi_{n+1}(x, y)$ is continuously differentiable and its derivative is bounded by an integrable function $g_{n+1}(x)$.

A3. For any $y \in \mathbb{R}$, the mapping $x \to \Phi_{n+1}(x, y)$ is continuously differentiable and its total variation $\mathrm{TV}(\Phi_{n+1}(\cdot, y))$ is bounded by a constant M_{n+1} (independent of y). We denote $\Phi_{n+1}(+\infty, y) := \lim_{x \to +\infty} \Phi_{n+1}(x, y)$.

Proposition 1 *For the classical Monte Carlo method, the following holds.*

1. *For any* $s \in \mathcal{B}_+(\mathbb{R})$,
$$\mathbb{E}\left[\frac{1}{N} \sum_{k=1}^{N} s(X_k^n)\right] = \int_{\mathbb{R}} s(x) d P_n(x).$$
2. *For any* $w \in \mathbb{R}$,
$$\mathrm{Var}\left(\frac{1}{N} \sum_{k=1}^{N} s_w(X_k^n)\right) \leq \frac{\alpha_n}{N},$$
where $\alpha_{n+1} = M_{n+1}^2 \alpha_n + 1/4$ $(n \geq 0)$.

Proof 1. We prove the result by induction. The claim holds for $n = 0$ from (5). For an arbitrary $n \geq 0$,

$$D_s^{n+1} := \int_{\mathbb{R}} s(x) dP_{n+1}(x) - \frac{1}{N} \sum_{k=1}^{N} s(X_k^{n+1}) = D_{s,1}^n + D_{s,2}^n,$$

where

$$D_{s,1}^n := \int_{\mathbb{R}} \int_I s \circ \varphi_{n+1}(x, u) du dP_n(x) - \frac{1}{N} \sum_{k=1}^{N} \int_I s \circ \varphi_{n+1}(X_k^n, u) du$$

and

$$D_{s,2}^n := \frac{1}{N} \sum_{k=1}^{N} \int_I s \circ \varphi_{n+1}(X_k^n, u) du - \frac{1}{N} \sum_{k=1}^{N} s \circ \varphi_{n+1}(X_k^n, U_k^{n+1}).$$

Since $\mathbb{E}[D_{s,1}^n] = 0$ by the induction hypothesis and $\mathbb{E}[D_{s,2}^n] = 0$ by item 1 of Lemma 1, the result holds.

2. We proceed by induction on n, with the case $n = 0$ being given by (6). Let $n \geq 0$ be arbitrary. As $\{X_1^n, X_2^n, \ldots, X_N^n\}$ and $\{U_1^{n+1}, U_2^{n+1}, \ldots, U_N^{n+1}\}$ are independent, we have also $\mathbb{E}[D_{s_w,1}^n D_{s_w,2}^n] = 0$. Hence

$$\text{Var}\left(\frac{1}{N} \sum_{k=1}^{N} s_w(X_k^{n+1})\right) = \mathbb{E}[(D_{s_w}^{n+1})^2] = \mathbb{E}[(D_{s_w,1}^n)^2] + \mathbb{E}[(D_{s_w,2}^n)^2]. \tag{9}$$

For the first summand, we write

$$\begin{aligned} \int_{\mathbb{R}\times I} s_w \circ \varphi_{n+1}(x, u) du dP_n(x) &= \int_{\mathbb{R}} \Phi_{n+1}(x, w) dP_n(x) \\ &= \Phi_{n+1}(+\infty, w) - \int_{\mathbb{R}} \frac{\partial \Phi_{n+1}}{\partial x}(x, w) F_n(x) dx, \end{aligned}$$

and

$$\int_I s_w \circ \varphi_{n+1}(X_k^n, u) du = \Phi_{n+1}(X_k^n, w).$$

This gives

$$\begin{aligned} D_{s_w,1}^n &= \int_{\mathbb{R}} \left(\int_{\mathbb{R}} s_x(y) dP_n(y) - \frac{1}{N} \sum_{k=1}^{N} s_x(X_k^n) \right) \left(-\frac{\partial \Phi_{n+1}}{\partial x}(x, w) \right) dx \\ &= \int_{\mathbb{R}} D_{s_x}^n \left(-\frac{\partial \Phi_{n+1}}{\partial x}(x, w) \right) dx. \end{aligned}$$

Due to the first item of this proposition, we may write

$$\begin{aligned}\mathbb{E}[(D^n_{s_w,1})^2] &= \int_{\mathbb{R}^2} \mathbb{E}\left[D^n_{s_x} D^n_{s_{x'}}\right] \frac{\partial \Phi_{n+1}}{\partial x}(x,w)\frac{\partial \Phi_{n+1}}{\partial x}(x',w)dxdx' \\ &\le \int_{\mathbb{R}^2} \sigma\left(\frac{1}{N}\sum_{k=1}^N s_x(X^n_k)\right)\sigma\left(\frac{1}{N}\sum_{k=1}^N s_{x'}(X^n_k)\right) \\ &\quad \cdot \left|\frac{\partial \Phi_{n+1}}{\partial x}(x,w)\frac{\partial \Phi_{n+1}}{\partial x}(x',w)\right| dxdx' \\ &\le (\mathrm{TV}(\Phi_{n+1}(\cdot,w)))^2 \sup_{x\in\mathbb{R}} \mathrm{Var}\left(\frac{1}{N}\sum_{k=1}^N s_x(X^n_k)\right).\end{aligned}$$

For the second summand in (9), item 2 of Lemma 1 implies:

$$\mathbb{E}[(D^n_{s_w,2})^2] \le \frac{1}{4N}.$$

By using (9), the result is established by induction.

□

3.2 Simple Stratified Sampling

As for classical Monte Carlo, we first focus on one step of the algorithm and we assume that $y^n_1, y^n_2, \ldots, y^n_N$ are given numbers. We suppose now that $y^n_1 \le y^n_2 \le \cdots \le y^n_N$. We consider the estimator $\widehat{Y}^{n+1}_s$ of $\int_{\mathbb{R}} s(x)d\widetilde{P}_{n+1}(x)$.

Lemma 2 *For the simple stratified sampling method, we have:*

1. *For any* $s \in \mathcal{B}_+(\mathbb{R})$,
$$\mathbb{E}[\widehat{Y}^{n+1}_s] = \int_{\mathbb{R}} s(x)d\widetilde{P}_{n+1}(x).$$
2. *For any* $w \in \mathbb{R}$,
$$\mathrm{Var}(\widehat{Y}^{n+1}_{s_w}) \le \frac{M_{n+1}+2}{4N^{3/2}}.$$

Proof 1. We have

$$\begin{aligned}\mathbb{E}[\widehat{Y}^{n+1}_s] &= \sum_{\ell\in[1,p]^2}\sum_{k=1}^N \int_{J_\ell} 1_k(v_{\ell,1}) s\circ\varphi_{n+1}(y^n_k, v_{\ell,2})dv_\ell \\ &= \frac{1}{N}\sum_{k=1}^N \int_I s\circ\varphi_{n+1}(y^n_k, v_2)dv_2,\end{aligned}$$

and the result follows from (3).

2. The function $C^n_{s_w}$ is the indicator function of the set

$$\mathcal{J}^n_w := \bigcup_{k=1}^{N} H_k \times [0, \Phi_{n+1}(y^n_k, w)). \tag{10}$$

The variable $C^n_{s_w}(V^{n+1}_\ell)$ is a Bernoulli random variable, with expectation $e^{n+1}_{w,\ell} = N\lambda_2(\mathcal{J}^n_w \cap J_\ell)$. Hence, $\mathrm{Var}(C^n_{s_w}(V^{n+1}_\ell)) \le 1/4$ and $\mathrm{Var}(C^n_{s_w}(V^{n+1}_\ell)) = 0$ if $J_\ell \subset \mathcal{J}^n_w$ or if $J_\ell \cap \mathcal{J}^n_w = \emptyset$. Therefore,

$$\mathrm{Var}(\widehat{Y}^{n+1}_{s_w}) \le \frac{1}{4N^2} \left|\{\ell \in [1, p]^2 : J_\ell \not\subset \mathcal{J}^n_w \text{ and } J_\ell \cap \mathcal{J}^n_w \neq \emptyset\}\right|.$$

By the same reasoning as in the proof of Lemma 2 in [4], we obtain the bounds

$$\begin{aligned}
&\left|\{\ell \in [1, p]^2 : J_\ell \not\subset \mathcal{J}^n_w \text{ and } J_\ell \cap \mathcal{J}^n_w \neq \emptyset\}\right| \\
&\quad \le p \sum_{\ell_1=1}^{p} \left(\max_{p(\ell_1-1)<k\le p\ell_1} \Phi_{n+1}(y^n_k, w) - \min_{p(\ell_1-1)<k\le p\ell_1} \Phi_{n+1}(y^n_k, w) \right) + 2p \\
&\quad \le p\mathrm{TV}(\Phi_{n+1}(\cdot, w)) + 2p, \qquad (11)
\end{aligned}$$

because we have $y^n_1 \le y^n_2 \le \cdots \le y^n_N$. The result follows.

□

The proof of the next result is similar to the proof of Proposition 1.

Proposition 2 *For the simple stratified sampling method, the following holds.*

1. *For any $s \in \mathcal{B}_+(\mathbb{R})$,*

$$\mathbb{E}\left[\frac{1}{N}\sum_{k=1}^{N} s(Y^n_k)\right] = \int_{\mathbb{R}} s(x) d P_n(x).$$

2. *For any $w \in \mathbb{R}$,*

$$\mathrm{Var}\left(\frac{1}{N}\sum_{k=1}^{N} s_w(Y^n_k)\right) \le \frac{\beta_n}{N^{3/2}},$$

where $\beta_{n+1} = M^2_{n+1}\beta_n + (M_{n+1} + 2)/4$ $(n \ge 0)$.

3.3 Sudoku Latin Square Sampling

As above, we first assume that $z_1^n, z_2^n, \ldots, z_N^n$ are given numbers and we suppose that $z_1^n \le z_2^n \le \cdots \le z_N^n$. We consider the estimator $\widehat{Z}_s^{n+1}$ of $\int_{\mathbb{R}} s(x) d\widetilde{P}_{n+1}(x)$. We make the following additional assumption.

A4. For any $y \in \mathbb{R}$, the mapping $x \to \Phi_{n+1}(x, y)$ is a piecewise monotonic function, with r_{n+1} pieces (this number is independent of y).

It then holds that $M_{n+1} \le r_{n+1}$.

Lemma 3 *For the Sudoku Latin square sampling method, we have:*

1. *For any $s \in \mathcal{B}_+(\mathbb{R})$,*
$$\mathbb{E}[\widehat{Z}_s^{n+1}] = \int_{\mathbb{R}} s(x) d\widetilde{P}_{n+1}(x).$$

2. *For any $w \in \mathbb{R}$,*
$$\mathrm{Var}(\widehat{Z}_{s_w}^{n+1}) \le (M_{n+1} + 2)\left(2M_{n+1} + r_{n+1} + \frac{29}{4}\right)\frac{1}{N^{3/2}}.$$

Proof 1. Since $W_\ell^{n+1} \sim \mathcal{U}(J_\ell)$, the argument is the same as in Lemma 2.

2. In the following, we encounter several summations with indexes $\ell, \ell', m, m' \in [1, p]^2$. To simplify notation, we omit writing $[1, p]^2$. We have
$$\mathrm{Var}(\widehat{Z}_{s_w}^{n+1}) = V_0(\widehat{Z}_{s_w}^{n+1}) + \frac{1}{N^2}\sum_{(\ell,\ell'):\ell\neq\ell'} \mathrm{Cov}\left(C_{s_w}^n(W_\ell^{n+1}), C_{s_w}^n(W_{\ell'}^{n+1})\right), \quad (12)$$
where
$$V_0(\widehat{Z}_{s_w}^{n+1}) := \frac{1}{N^2}\sum_{\ell} \mathrm{Var}\left(C_{s_w}^n(W_\ell^{n+1})\right).$$
Since $W_\ell^{n+1} \sim \mathcal{U}(J_\ell)$ and the states are ordered, $z_1^n \le z_2^n \le \cdots \le z_N^n$, we have, as in Lemma 2:
$$V_0(\widehat{Z}_{s_w}^{n+1}) \le \frac{M_{n+1} + 2}{4N^{3/2}}.$$
We split $\mathrm{Var}(\widehat{Z}_{s_w}^{n+1}) = V_0(\widehat{Z}_{s_w}^{n+1}) + V_1(\widehat{Z}_{s_w}^{n+1}) + V_2(\widehat{Z}_{s_w}^{n+1}) + V_3(\widehat{Z}_{s_w}^{n+1})$, with
$$V_1(\widehat{Z}_{s_w}^{n+1}) := \frac{1}{N^2}\sum_{(\ell,\ell'):\ell_1\neq\ell_1'\wedge\ell_2=\ell_2'} \mathrm{Cov}\left(C_{s_w}^n(W_\ell^{n+1}), C_{s_w}^n(W_{\ell'}^{n+1})\right),$$

$$V_2(\widehat{Z}_{s_w}^{n+1}) := \frac{1}{N^2} \sum_{(\ell,\ell'):\ell_1=\ell_1' \wedge \ell_2 \neq \ell_2'} \mathrm{Cov}\left(C_{s_w}^n(W_\ell^{n+1}), C_{s_w}^n(W_{\ell'}^{n+1})\right),$$

$$V_3(\widehat{Z}_{s_w}^{n+1}) := \frac{1}{N^2} \sum_{(\ell,\ell'):\ell_1\neq\ell_1' \wedge \ell_2 \neq \ell_2'} \mathrm{Cov}\left(C_{s_w}^n(W_\ell^{n+1}), C_{s_w}^n(W_{\ell'}^{n+1})\right).$$

We introduce the N^2 squares $J_{\ell,m} = H_{\ell_1,m_1} \times H_{\ell_2,m_2}$, for $(\ell, m) \in [1, p]^4$, where

$$H_{\ell_i,m_i} := [(\ell_i - 1)/p + (m_i - 1)/N, (\ell_i - 1)/p + m_i/N), \text{ for } i = 1, 2.$$

If $\mathcal{J}_w^n$ is defined as in (10) (replacing the y_k^n with the z_k^n), the same reasoning as in Lemma 2 leads to

$$V_1(\widehat{Z}_{s_w}^{n+1}) = \sum_{\ell: J_\ell \not\subset \mathcal{J}_w^n \wedge J_\ell \cap \mathcal{J}_w^n \neq \emptyset} \sum_{\ell': \ell_1' \neq \ell_1 \wedge \ell_2' = \ell_2} V_1(\ell, \ell'),$$

where

$$V_1(\ell, \ell') := \frac{N}{p-1} \sum_{(m,m'):m_1=m_1' \wedge m_2 \neq m_2'} \lambda_2(J_{\ell,m} \cap \mathcal{J}_w^n)\lambda_2(J_{\ell',m'} \cap \mathcal{J}_w^n) - \lambda_2(J_\ell \cap \mathcal{J}_w^n)\lambda_2(J_{\ell'} \cap \mathcal{J}_w^n).$$

Similarly,

$$V_2(\widehat{Z}_{s_w}^{n+1}) = \sum_{\ell: J_\ell \not\subset \mathcal{J}_w^n \wedge J_\ell \cap \mathcal{J}_w^n \neq \emptyset} \sum_{\ell': \ell_1' = \ell_1 \wedge \ell_2' \neq \ell_2} V_2(\ell, \ell')$$

with

$$V_2(\ell, \ell') := \frac{N}{p-1} \sum_{(m,m'):m_1\neq m_1' \wedge m_2 = m_2'} \lambda_2(J_{\ell,m} \cap \mathcal{J}_w^n)\lambda_2(J_{\ell',m'} \cap \mathcal{J}_w^n) - \lambda_2(J_\ell \cap \mathcal{J}_w^n)\lambda_2(J_{\ell'} \cap \mathcal{J}_w^n)$$

and

$$V_3(\widehat{Z}_{s_w}^{n+1}) = \sum_{\ell: J_\ell \not\subset \mathcal{J}_w^n \wedge J_\ell \cap \mathcal{J}_w^n \neq \emptyset} \sum_{\substack{\ell': \ell_1' \neq \ell_1 \wedge \ell_2' \neq \ell_2 \\ J_{\ell'} \not\subset \mathcal{J}_w^n \wedge J_{\ell'} \cap \mathcal{J}_w^n \neq \emptyset}} V_3(\ell, \ell'),$$

where

$$V_3(\ell, \ell') := \frac{N}{(p-1)^2} \sum_{(m,m'):m_1\neq m_1' \wedge m_2 \neq m_2'} \lambda_2(J_{\ell,m} \cap \mathcal{J}_w^n)\lambda_2(J_{\ell',m'} \cap \mathcal{J}_w^n) - \lambda_2(J_\ell \cap \mathcal{J}_w^n)\lambda_2(J_{\ell'} \cap \mathcal{J}_w^n).$$

We then follow the technical steps of the proof of Lemma 3 in [4] with the set $\mathcal{J}_w^n$. By using (11), we obtain the bounds:

$$|V_1(\widehat{Z}_{s_w}^{n+1})| \leq (M_{n+1}+2)(p(r_{n+1}+1)-1)\frac{1}{N^2},$$
$$|V_2(\widehat{Z}_{s_w}^{n+1})| \leq (M_{n+1}+2)(2p-1)\frac{1}{N^2},$$
$$|V_3(\widehat{Z}_{s_w}^{n+1})| \leq (M_{n+1}+2)^2(2p-1)\frac{1}{N^2}.$$

This proves the result.

□

The proof of the next result is similar to the proof of Proposition 1.

Proposition 3 *For the Sudoku Latin square sampling method, the following holds.*

1. *For any* $s \in \mathcal{B}_+(\mathbb{R})$,

$$\mathbb{E}\left[\frac{1}{N}\sum_{k=1}^{N} s(Z_k^n)\right] = \int_{\mathbb{R}} s(x)dP_n(x).$$

2. *For any* $w \in \mathbb{R}$,

$$\mathrm{Var}\left(\frac{1}{N}\sum_{k=1}^{N} s_w(Z_k^n)\right) \leq \frac{\gamma_n}{N^{3/2}},$$

where $\gamma_{n+1} = M_{n+1}^2\gamma_n + (M_{n+1}+2)(2M_{n+1}+r_{n+1}+29/4)$ $(n \geq 0)$.

Remark 1 The constant involved in the $\mathcal{O}(N^{-3/2})$ bound of $\mathrm{Var}(\widehat{Z}_{s_w}^{n+1})$ (SLSS) is larger than the corresponding constant for $\mathrm{Var}(\widehat{Y}_{s_w}^{n+1})$ (SSS). If $\beta_0 = \gamma_0$ (which is satisfied if the Y_k^0 and Z_k^0 are generated by inversion), this would suggest poorer performance of SLSS choice, because then $\beta_n \leq \gamma_n$ for any n. It is not the case in the examples of Sect. 4 below. For bounding $\mathrm{Var}(\widehat{Z}_{s_w}^{n+1})$, we use (12) and we bound the absolute values of the covariances, without consideration of the signs. If some covariances are negative, our upper bound of the variance is too loose.

4 Numerical Experiments

In this section, we compare the three approaches for the simulation of Markov chains that we have analyzed: Monte Carlo, simple stratified sampling and Sudoku Latin square sampling. We calculate the empirical variance of the estimators of

$$\int_{\mathbb{R}} s(x)dP_n(x)$$

that we have defined above, for some s. We plot this variance as a function of the number N of simulated chains. If we assume a model $KN^{-\delta}$ for the variance, the rate δ can be estimated by linear regression and can be compared with the theoretical bounds. We also compute the *efficiency* of each method, defined as the inverse of the product of the variance and the CPU time [14]. For standard MC, the efficiency does not depend on N. On the other hand, sorting the chains at each step of the stratified methods brings additional overhead. It requires an $\mathcal{O}(N \log N)$ effort, while advancing the chains (excluding the renumbering) requires $\mathcal{O}(N)$ time, as for MC.

In what follows, we denote by φ and Φ the density and the cumulative distribution function of the standard normal distribution, respectively. The notation $X \sim \mathcal{N}(0, 1)$ means that the random variable X has density φ.

4.1 An Autoregressive Process

We consider the same simple autoregressive process of order one as in [15]. Let us define a Markov chain over $\mathbb{R}$ which evolves according to the recurrence:

$$Y_1 = Z_1 \quad \text{and} \quad Y_{n+1} = \frac{1}{\sqrt{\beta^2 + 1}}(\beta Y_n + Z_{n+1}), \quad n \geq 1,$$

where $\beta \geq 0$ is a constant and $Z_1, Z_2, \ldots$ are i.i.d. standard normal random variables. We have $Y_n \sim \mathcal{N}(0, 1)$ for every n. By setting $X_n := \Phi(Y_n)$ and $U_n := \Phi(Z_n)$, the recurrence may be written as in (1):

$$X_1 = U_1 \quad \text{and} \quad X_{n+1} = \Phi\left(\frac{1}{\sqrt{\beta^2 + 1}}(\beta \Phi^{-1}(X_n) + \Phi^{-1}(U_{n+1})\right), \quad n \geq 1,$$

where $U_1, U_2, \ldots$ are i.i.d. uniform random variables over I. We have $X_n \sim \mathfrak{U}(I)$ for every n. We take $\beta = 0.1$. We estimate the probability that $X_n > 0.5$ for $n = 100$. We replicate the calculation independently 500 times and we compute the sample variance. Figure 2 shows the $\log_{10}$ of the variance as a function of $\log_{10} N$ on the left and the $\log_{10}$ of the efficiency as a function of $\log_{10} N$ on the right, for $N = 10^2, 20^2, \ldots, 300^2$.

We find from the plots that SSS and SLSS give not only smaller variances than standard MC (for the same N), but also better efficiencies, and that SLSS is slightly superior to SSS. The regression estimates of δ are given in the first row of Table 1. They match the rates $\mathcal{O}(N^{-1})$ and $\mathcal{O}(N^{-3/2})$ established in Sect. 3 for MC and SSS (or SLSS), respectively. This is a very simple problem since all the X_n are uniform random variables over I. The variances of the stratification strategies are smaller than those of the standard MC, even when few copies of the chain are simulated, and the additional overhead due to sorting does not balance out the gain in variance.

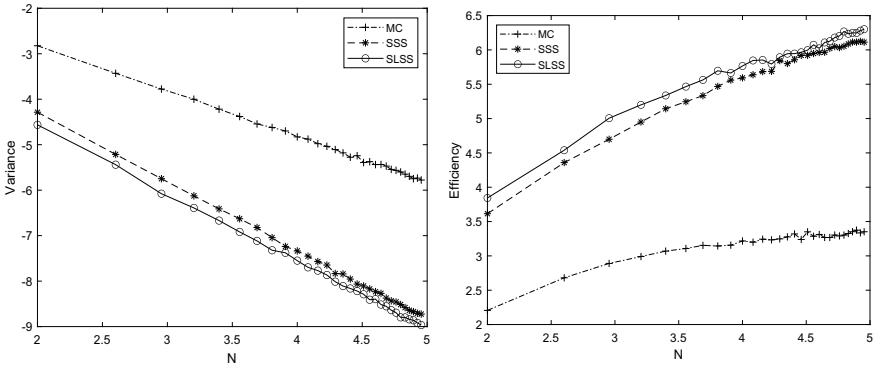


Fig. 2 Autoregressive process: sample variance (left) and efficiency (right) of 500 copies of the calculation of $\mathbb{P}(X_{100} > 0.5)$ as a function of N ($N = 10^2, 20^2, \ldots, 300^2$) ($\log_{10}$-$\log_{10}$ scale)

4.2 A European Put Option

In the Black-Scholes model, under the risk-neutral measure, the asset price S_t at time t is given by:

$$S_t = S_0 \exp((r - \sigma^2/2)t + \sigma B_t), \tag{13}$$

where r is the risk-free interest rate, σ the volatility parameter and B is a standard Brownian motion. Let T be the maturity date and K the strike price. We want to estimate the value of the put option, $PO := \mathrm{e}^{-rT}\mathbb{E}[(K - S_T)_+]$. To formulate the problem as a Markov chain, we discretize the interval $[0, T]$ using observation times $0 = t_0 < t_1 < \cdots < t_N = T$. The discrete version of (13) can be written as:

$$S_{t_{n+1}} = S_{t_n} \exp((r - \sigma^2/2)\Delta t_{n+1} + \sigma(B_{t_{n+1}} - B_{t_n})), \quad n \geq 0,$$

where $\Delta t_{n+1} := t_{n+1} - t_n$. By setting $X_n := S_{t_n}$, this may be written as in (1):

$$X_{n+1} = X_n \exp((r - \sigma^2/2)\Delta t_{n+1} + \sigma\sqrt{\Delta t_{n+1}}\Phi^{-1}(U_{n+1})), \quad n \geq 0,$$

where $U_1, U_2, \ldots$ are i.i.d. uniform random variables over I. This example is somehow artificial: (1) the value of PO is given explicitly by the Black-Scholes formula, so there is no need to estimate it; (2) if simulation is used, the value S_T can be generated directly and there is no need to generate the intermediate values. Nevertheless, the object comes from real life. We choose the following parameters: $S_0 = 100$, $K = 120$, $r = 0.1$, $\sigma = 0.2$, $T = 1$ and $\Delta t_{n+1} = T/P$, with $P = 100$. We compare the variances of the MC, SSS and SLSS estimators of PO: we replicate the calculation independently 500 times and we compute the sample variance. Figure 3 shows $\log_{10}$ of the variance as a function of $\log_{10} N$ on the left and $\log_{10}$ of the efficiency as a function of $\log_{10} N$ on the right, for $N = 10^2, 20^2, \ldots, 300^2$.

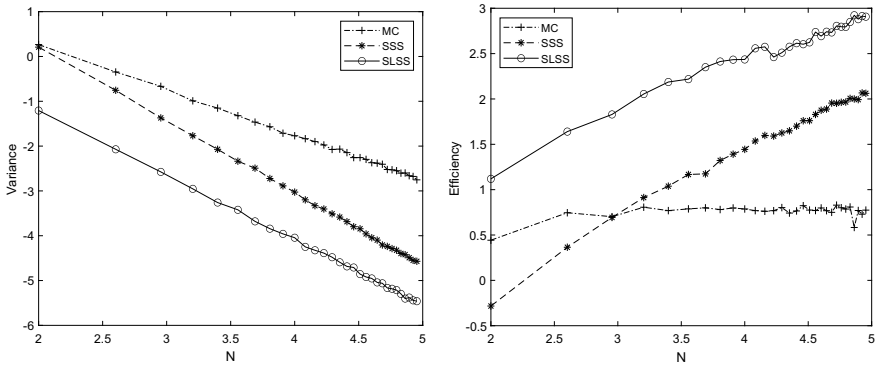


Fig. 3 European put option: sample variance (left) and efficiency (right) of 500 copies of the calculation of $e^{-rT}\mathbb{E}[(K - S_T)_+]$ as a function of N ($N = 10^2, 20^2, \ldots, 300^2$) ($\log_{10}$-$\log_{10}$ scale)

We clearly see that SSS and SLSS produce smaller variances than MC. When comparing the results of SSS and SLSS, we see that the later approach outperforms the former. The regression estimates of δ are given in the second row of Table 1. They are comparable to the rates $\mathcal{O}(N^{-1})$ and $\mathcal{O}(N^{-3/2})$ of the upper bounds demonstrated in Sect. 3 for MC and SSS (or SLSS), respectively. Here, the additional sorting times cause a degraded efficiency of SSS when $N \leq 900$. The better convergence rate of the stratified approach reestablishes a superior efficiency when the number of copies is increased.

An example with a call option is described in [8].

4.3 Diffusion

We consider the initial value problem for the heat equation:

$$\frac{\partial c}{\partial t}(x,t) = D\frac{\partial^2 c}{\partial x^2}(x,t), \quad x \in \mathbb{R},\ t > 0 \quad \text{and} \quad c(x,0) = c_0(x), \quad x \in \mathbb{R},$$

with constant diffusion coefficient $D > 0$. We assume that the initial data satisfies $c_0 \geq 0$ and $\int_{\mathbb{R}} c_0(x)dx = 1$. Then, for any $t > 0$, it holds that $\int_{\mathbb{R}} c(x,t)dx = 1$. Let G be the fundamental solution of the heat operator:

$$G(x,t) := \frac{1}{\sqrt{4\pi Dt}} e^{-x^2/4Dt}, \quad x \in \mathbb{R},\ t > 0.$$

Then, for any $\tau \geq 0$,

$$c(x,t) = \int_{\mathbb{R}} G(x-w, t-\tau)c(w,\tau)dw, \quad x \in \mathbb{R},\ t > \tau.$$

If Δt is a time step, we denote $t_n := n\Delta t$ and $c_n(x) := c(x, t_n)$. It follows that

$$c_{n+1}(x) = \int_{\mathbb{R}} G(x-w, \Delta t)c_n(w)dw = \frac{1}{\sqrt{2D\Delta t}} \int_{\mathbb{R}} \varphi\left(\frac{x-w}{\sqrt{2D\Delta t}}\right) c_n(w)dw.$$

Consequently, for any $s \in \mathcal{B}_+(\mathbb{R})$,

$$\int_{\mathbb{R}} s(x)c_{n+1}(x)dx = \int_{\mathbb{R}\times I} s(x + \sqrt{2D\Delta t}\Phi^{-1}(u))c_n(x)dxdu.$$

We define the following Markov chain. Let X_0 have probability density function c_0 and let

$$X_{n+1} = X_n + \sqrt{2D\Delta t}\Phi^{-1}(U_{n+1}),$$

where $U_{n+1} \sim \mathfrak{U}(I)$. This defines a *random walk* method: see [10] and [12, 19] in a QMC context. As presented, the algorithm is artificial, since we know the exact solution. This method is a part of fractional step schemes when we consider problems involving a combination of convection, reaction and diffusion. We take $D = 1$ and define c_0 as the indicator function of the interval $[-1/2, 1/2]$. We choose $\Delta t = 1/100$ and $T = 1$. We compute the empirical variance (with $M = 500$ replications) of the estimate of $\int_a^b c(x, T)dx$, with $a = 2, b = 4$. Figure 4 shows $\log_{10}$ of the variance as a function of $\log_{10} N$ on the left and $\log_{10}$ of the efficiency as a function of $\log_{10} N$ on the right, for $N = 10^2, 20^2, \ldots, 300^2$.

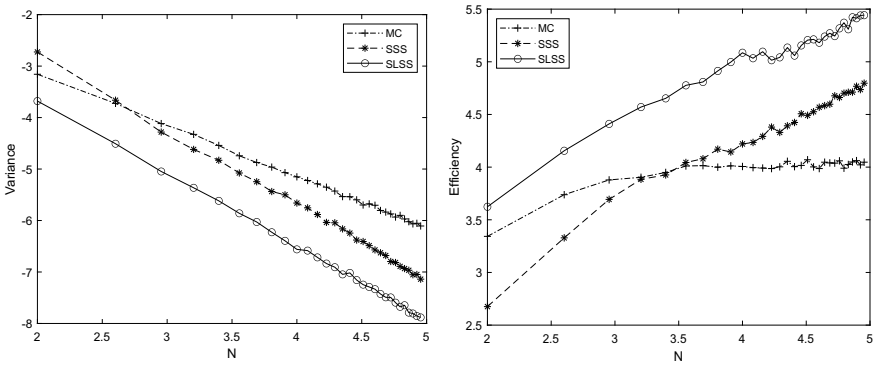


Fig. 4 Diffusion: sample variance (left) and efficiency (right) of 500 copies of the calculation of $\int_2^4 c(x, T)dx$ as a function of N ($N = 10^2, 20^2, \ldots, 300^2$) ($\log_{10}$-$\log_{10}$ scale)

Table 1 Estimation of the convergence rate δ of the sample variance: comparison of standard MC, SSS, and SLSS for three examples and estimators

Experiment	Quantity of interest	MC	SSS	SLSS
Autoregressive process	$\mathbb{P}(X_{100} > 0.5)$	1.00	1.51	1.48
European put option	$e^{-rT}\mathbb{E}[(K - S_T)_+]$	1.01	1.62	1.44
Diffusion	$\int_2^4 c(x, T)dx$	1.00	1.46	1.43

The SLSS approach gives smaller variances than classical MC (for the same N), and better efficiencies. In addition, SLSS is superior to SSS. The regression estimates of δ are given in the third row of Table 1. They are near the orders of $\mathcal{O}(N^{-1})$ and $\mathcal{O}(N^{-3/2})$ established in Sect. 3 for MC and SSS (or SLSS), respectively. One must notice that our bounds do not prove that the SSS and SLSS variances are smaller than the standard MC variance, for any N. It may be hoped that, if our bounds are tight, then the stratification variances are below the MC variance, for large enough $N's$. In this experiment, the variance of the SSS method is larger than the MC variance, when the number of simulated copies is $\leq$400. This entails a lower efficiency. The superiority of the SSS approach only appears when more than 3600 copies are used. Note that the growth regime of the efficiency also depends on the programming language and computer used.

Other experiments are reported in [5] and also in [4], where a supplementary spatial step is used.

5 Conclusions

We consider stratified MC methods for Markov chain models with a one-dimensional continuous state space. We assume that only one random variate is used to advance the chain by one step. We provide upper bounds on the variance under different sampling methods: ordinary MC and two stratified approaches, SSS and SLSS. When N copies of the chain are simulated, the order is $\mathcal{O}(N^{-1})$ for MC and $\mathcal{O}(N^{-3/2})$ for SSS and SLSS. This analysis complements a previous study for Markov chains with discrete state spaces, where similar bounds are established. In numerical examples, the variance rates match the orders of the theoretical upper bounds and we observe that SLSS give smaller variance than SSS. The extension of the analysis to multi-dimensional problems will be the subject of future work.

References

1. Abdallah, B.A., L'Ecuyer, P., Puchhammer, F.: Array-RQMC for option pricing under stochastic volatility models. In: Mustafee, N., Bae, K.H.G., Lazarova-Molnar, S., Rabe, M., Szabo, G., Haas, P., Son, Y.J. (eds.) Proceedings of the 2019 Winter Simulation Conference, pp. 440–451. IEEE Press (2019)
2. Cheng, R.C.H., Davenport, T.: The problem of dimensionality in stratified sampling. Manage. Sci. **35**, 1278–1296 (1989)
3. Dick, J., Pillichshammer, F.: Digital Nets and Sequences. Cambridge University Press, Cambridge (2010)
4. El Haddad, R., El Maalouf, J., Lécot, C., L'Ecuyer, P.: Sudoku Latin square sampling for Markov chain simulation. In: Tuffin, B., L'Ecuyer, P. (eds.) Monte Carlo and Quasi-Monte Carlo Methods, pp. 207–230. Springer, Cham (2020)
5. El Haddad, R., Fakhereddine, R., Lécot, C., Venkiteswaran, G.: Extended Latin hypercube square sampling for integration and simulation. In: Dick, J., Kuo, F.Y., Peters, G.W., Sloan, I.H. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2012, pp. 317–330. Springer, Berlin (2013)
6. El Haddad, R., Lécot, C., L'Ecuyer, P.: Quasi-Monte Carlo simulation of discrete-time Markov chains on multidimensional state spaces. In: Keller, A., Heinrich, S., Niederreiter, H. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2006, pp. 413–429. Springer, Berlin (2008)
7. El Haddad, R., Lécot, C., L'Ecuyer, P., Nassif, N.: Quasi-Monte Carlo methods for Markov chains with continuous multi-dimensional state space. Math. Comput. Simul. **81**, 560–567 (2010)
8. Fakhereddine, R., El Haddad, R., Lécot, C., El Maalouf, J.: Stratified Monte Carlo simulation of Markov chains. Math. Comput. Simul. **135**, 51–62 (2017)
9. Gerber, M., Chopin, N.: Sequential quasi-Monte Carlo. J. Roy. Stat. Soc. B **77**, 509–579 (2015)
10. Ghoniem, A.F., Sherman, F.S.: Grid-free simulation of diffusion using random walk methods. J. Comput. Phys. **61**, 1–37 (1985)
11. Haber, S.: A modified Monte-Carlo quadrature. Math. Comput. **20**, 361–368 (1966)
12. Lécot, C., El Khettabi, F.: Quasi-Monte Carlo simulation of diffusion. J. Complex. **15**, 342–359 (1999)
13. Lécot, C., Tuffin, B.: Quasi-Monte Carlo methods for estimating transient measures of discrete time Markov chains. In: Niederreiter, H. (ed.) Monte Carlo and Quasi-Monte Carlo Methods 2002, pp. 329–343. Springer, Berlin (2004)
14. L'Ecuyer, P.: Efficiency improvement and variance reduction. In: Tew, J.D., Manivannan, S., Sadowski, D.A., Seila, A.F. (eds.) Proceedings of the 1994 Winter Simulation Conference, pp. 122–132. IEEE Press (1994)
15. L'Ecuyer, P., Lécot, C., L'Archevêque-Gaudet, A.: On Array-RQMC for Markov chains: mapping alternatives and convergence rates. In: L'Ecuyer, P., Owen, A. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2008, pp. 485–500. Springer, Berlin (2009)
16. L'Ecuyer, P., Lécot, C., Tuffin, B.: Randomized quasi-Monte Carlo simulation of Markov chains with an ordered state space. In: Niederreiter, H., Talay, D. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2004, pp. 331–342. Springer, Berlin (2006)
17. L'Ecuyer, P., Lécot, C., Tuffin, B.: A randomized quasi-Monte Carlo simulation method for Markov chains. Oper. Res. **56**, 958–975 (2008)
18. L'Ecuyer, P., Munger, D., Lécot, C., Tuffin, B.: Sorting methods and convergence rates for Array-RQMC: Some empirical comparisons. Math. Comput. Simul. **143**, 191–201 (2018)
19. Morokoff, W.J., Caflisch, R.E.: A Quasi-Monte Carlo approach to particle simulation of the heat equation. SIAM J. Numer. Anal. **30**, 1558–1573 (1993)
20. Niederreiter, H.: Random Number Generation and Quasi-Monte Carlo Methods. SIAM, Philadelphia, Pennsylvania (1992)
21. Owen, A.B.: Orthogonal arrays for computer experiments, integration and visualization. Stat. Sin. **2**, 439–452 (1992)

22. Owen, A.B.: Lattice sampling revisited: Monte Carlo variance of means over randomized orthogonal arrays. Ann. Stat. **22**, 930–945 (1994)
23. Owen, A.B.: Monte Carlo variance of scrambled net quadrature. SIAM J. Numer. Anal. **34**, 1884–1910 (1997)
24. Pedersen, R.M., Vis, T.L.: Sets of mutually orthogonal Sudoku Latin squares. Coll. Math. J. **40**, 174–180 (2009)
25. Puchhammer, F., Ben Abdellah, A., L'Ecuyer, P.: Variance reduction with Array-RQMC for tau-leaping simulation of stochastic biological and chemical reaction networks. Submitted
26. Tang, B.: Orthogonal array-based Latin hypercubes. J. Am. Stat. Assoc. **88**, 1392–1397 (1993)
27. Xu, X., Haaland, B., Qian, P.Z.G.: Sudoku-based space-filling designs. Biometrika **98**, 711–720 (2011)

Quasi-Random Sampling with Black Box or Acceptance-Rejection Inputs

Erik Hintz, Marius Hofert, and Christiane Lemieux

Abstract We propose randomized quasi-Monte Carlo (RQMC) methods to estimate expectations $\mu = \mathbb{E}(g(\boldsymbol{Y}, W))$ where $\boldsymbol{Y}$ is independent of W and can be sampled by inversion, whereas W cannot. Various practical problems are of this form, such as estimating expected shortfall for mixture models where W is stable or generalized inverse Gaussian and $\boldsymbol{Y}$ is multivariate normal. We consider two settings: In the first, we assume that there is a non-uniform random variate generation method to sample W in the form of a non-modifiable "black-box". The methods we propose for this setting are based on approximations of the quantile function of W. In the second setting, we assume that there is an acceptance-rejection (AR) algorithm to sample from W and explore different ways to feed it with quasi-random numbers. This has been studied previously, typically by rejecting points of constant dimension from a low-discrepancy sequence and moving along the sequence. We also investigate the use of a point set of constant (target) size where the dimension of each point is increased until acceptance. In addition, we show how to combine the methods from the two settings in such a way that the non-monotonicity inherent to AR is removed.

Keywords Quasi-random numbers · Black box · Acceptance-rejection · Normal variance mixtures

E. Hintz (✉) · M. Hofert · C. Lemieux
Department of Statistics and Actuarial Science, University of Waterloo,
200 University Avenue West, Waterloo, ON N2L 3G1, USA
e-mail: erik.hintz@uwaterloo.ca

M. Hofert
e-mail: marius.hofert@uwaterloo.ca

C. Lemieux
e-mail: clemieux@uwaterloo.ca

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_13

1 Introduction

Consider the problem of estimating the quantity

$$\mu = \mathbb{E}(g(\boldsymbol{Y}, W)) \tag{1}$$

where $g : \mathbb{R}^{d+1} \to \mathbb{R}$ is integrable and $\boldsymbol{Y} \sim F_{\boldsymbol{Y}}$ is a d-dimensional random vector independent of the random variable $W \sim F_W$. For instance, if $\boldsymbol{Y}$ is multivariate normal and W follows a generalized inverse Gaussian (GIG) distribution (see, e.g., [14] for an AR algorithm to sample from GIG distributions), we could be estimating the expected shortfall of a generalized hyperbolic distribution; this is an important class of multivariate distributions in risk management, see, e.g., [23].

The classical Monte Carlo (MC) estimator $\hat{\mu}_n^{\text{mc}}$ based on n samples for μ is given by

$$\hat{\mu}_n^{\text{mc}} = \frac{1}{n} \sum_{i=1}^{n} g(\boldsymbol{Y}_i, W_i),$$

where $(\boldsymbol{Y}_i, W_i) \overset{\text{ind.}}{\sim} F_{\boldsymbol{Y}} \times F_W$ for $i = 1, \dots, n$.

We assume that there is an easy way to sample from $F_{\boldsymbol{Y}}$ based on uniforms; e.g., based on the Rosenblatt transform [27]. That is to say, assume there is a transformation $T_{\boldsymbol{Y}} : (0,1)^{d+k} \to \mathbb{R}^d$ such that $T_{\boldsymbol{Y}}(\boldsymbol{U}) \sim F_{\boldsymbol{Y}}$ for $\boldsymbol{U} \sim \mathrm{U}^{d+k}$ for constant $k \geq 0$; if, e.g., $\boldsymbol{Y} \sim \mathbb{N}_d(\boldsymbol{\mu}, \Sigma)$ then $k = 0$ and the function $T_{\boldsymbol{Y}}(\boldsymbol{u})$ is given by $T_{\boldsymbol{Y}}(\boldsymbol{u}) = \boldsymbol{\mu} + A(\Phi^{-1}(u_1), \dots, \Phi^{-1}(u_d))^\top$ where A is such that $AA^\top = \Sigma$.

In this paper, we investigate the following question:

How can a randomized quasi-Monte Carlo (RQMC) estimator for μ be constructed when W cannot be sampled by inversion?

More precisely, we assume that the (always existing) quantile function $F_W^{\leftarrow}(u) = \inf\{x : F_W(x) \geq u\}$ is intractable and instead we rely on other methods for non-uniform random variate generation (NRVG), such as AR algorithms, where at first glance it may seem hard to directly apply RQMC methods.

We investigate the above question under two sets of assumptions on what we mean by the existence of a "NRVG" method for W.

1. Black-box case. Here, we assume that we have a (random) function $R_W : \mathbb{N} \to \mathbb{R}^n$ such that if $R_W(n) = \boldsymbol{W}$ for $\boldsymbol{W} = (W_1, \dots, W_n)$ then $W_i \overset{\text{ind.}}{\sim} F_W$ for $i = 1, \dots, n$. As such, we have a "black box" function that returns samples from F_W of any size. The underlying sampling method could be based on MCMC, machine learning techniques or methods based on a stochastic representation (SR), among others. In Sect. 2, we propose methods that estimate the quantile function $F_W^{\leftarrow}$, as well as re-ordering strategies that make the output of R_W mimick the behavior of the underlying LDS. We also perform a numerical study comparing our methods. We highlight the assumption that we have only access to R_W, irrespective of whether or not W admits a tractable density. If W does have a density that can be efficiently computed, other

methods that approximate the quantile function using this additional information may be better suited; see [5] for a popular method. There are, however, examples where this is not the case: if W follows a stable distribution, sampling is easy based on the stochastic representation derived in [1], but not even the density function can be computed without numerical integration.

2. *AR algorithms for W,* where the proposal (or envelope) distribution and the acceptance decision can be sampled by inversion of uniforms. The main difference to the black-box setting is that here, we do have access to the underlying sampling mechanism and can feed the AR sampler with a randomized low-discrepancy sequence (LDS). AR algorithms are typically not popular in RQMC as it is possibly infinite-dimensional. Smoothed rejection and weighted uniform sampling is considered in [24], along with numerical results showing that these outperform AR sampling in terms of convergence speed. It is shown in [28] that the F discrepancy, i.e., $\sup_x |F_n(x) - F(x)|$, where F_n and F denote the empirical and theoretical distribution function, of a sample obtained via AR is in $\mathcal{O}(n^{-\alpha})$ for $1/2 \leq \alpha < 1$. The error convergence rate is improved by replacing the purely binary AR decision with weights, called extended smoothed rejection. This circumvents integration of an indicator function. Discrepancy properties of points produced by totally deterministic AR methods, i.e., AR with a (non-randomized) Sobol' sequence are derived in [29]. A convergence result, error bounds and a numerical study for AR with RQMC is given in [26]. What all previous references have in common is that they hold the dimension of the LDS constant and effectively use a subset of size n of the first N points in the sequence. We investigate, among other things, whether there is a difference between holding d constant (and thereby skipping points in the sequence) or holding n constant (thereby thinking of the first n points having potentially unbounded dimension). This is the topic of Sect. 3.

To be clear, AR could even be an algorithm used within the black-box setting, but given its prevalence, we choose to treat AR separately. We revisit this point at the end of Sect. 3, where we combine ideas from both settings.

Section 4 applies the methods presented in Sects. 2 and 3 to the problem of estimating the price of a basket call option under a normal variance mixture copula dependence. As mixing distributions, we use the inverse-gamma distribution (as its known quantile function can be used as a benchmark) and the GIG distribution. In the latter case, we also include the method based on numerical inversion of the density in [5], which was shown to be efficient for the GIG in [21]. We perform the same experiment with the aforementioned stable mixture, a model where the method in [5] cannot be easily applied for sampling due to the lack of a tractable density. Section 5 concludes the paper.

2 Methods for the Black Box Setting

Recall that the classical MC estimator $\hat{\mu}_n^{\text{mc}}$ based on n samples for (1) can be written as

$$\hat{\mu}_n^{\text{mc}} = \frac{1}{n} \sum_{i=1}^{n} g(T_{\boldsymbol{Y}}(\boldsymbol{U}_i), W_i), \tag{2}$$

where $\boldsymbol{U}_i \overset{\text{ind.}}{\sim} \text{U}(0,1)^{d+k}$ is independent of $W_1, \ldots, W_n \overset{\text{ind.}}{\sim} F_W$ obtained by calling $R_W(n)$. To simplify the notation, we henceforth assume $k = 0$; the case $k > 0$ is handled by replacing d by $d + k$ in what follows. In order to be able to apply RQMC to the problem, we first rewrite (1) as an integral over the unit hypercube. With a change of variable, we obtain

$$\mu = \int_{(0,1)^{d+1}} g(T_{\boldsymbol{Y}}(u_{1:d}), Q(u_{d+1}))\, \mathrm{d}\boldsymbol{u}, \tag{3}$$

where $\boldsymbol{u} = (u_{1:d}, u_{d+1})$ with $u_{1:d} = (u_1, \ldots, u_d)$, and we use the function $Q : [0,1] \to \mathbb{R}$ as a shorthand notation for the quantile function $F_W^{\leftarrow}$ for the remainder of this paper.

If we were able to sample W via inversion, then RQMC sampling could be used to estimate μ using the following approach: Let $\tilde{P}_{b,n} = \{\boldsymbol{u}_{b,1}, \ldots, \boldsymbol{u}_{b,n}\} \subseteq [0,1)^{d+1}$, where $\boldsymbol{u}_{b,i} = (u_{b,i,1}, \ldots, u_{b,i,d+1})$ for $b = 1, \ldots, B$, denote B independent randomizations of the first n points of the low-discrepancy sequence (LDS) used; here we assume that the randomization is such that each $\boldsymbol{u}_{b,i} \sim U(0,1)^{d+1}$. Then

$$\hat{\mu}_{b,n}^{\text{rqmc}} = \frac{1}{n} \sum_{i=1}^{n} g(T_{\boldsymbol{Y}}(\boldsymbol{u}_{b,i,1:d}), Q(u_{b,i,d+1})), \quad b = 1, \ldots, B, \tag{4}$$

and an RQMC estimator for μ based on a total of nB points would be given by

$$\hat{\mu}_{B,n}^{\text{rqmc}} = \frac{1}{B} \sum_{b=1}^{B} \hat{\mu}_{b,n}^{\text{rqmc}}.$$

The variance/error of $\hat{\mu}_{B,n}^{\text{rqmc}}$ could then be estimated in the usual way; see [19].

However, we do not know Q, so the estimators $\hat{\mu}_{b,n}^{\text{rqmc}}$ in (4) cannot be computed. In this section, we propose two different methods to *approximate* $\hat{\mu}_{b,n}^{\text{rqmc}}$ for $b = 1, \ldots, B$. Both methods essentially replace Q by an estimate thereof.

2.1 Methods Based on the Empirical Quantile Function

A simple ad-hoc method to approximate $\hat{\mu}_{b,n}^{\text{rqmc}}$ could be to replace the Q values by a random sample of F_W obtained by calling $R_W(Bn)$. More precisely, let $W_{b,i}$ for $b = 1, \ldots, B$, $i = 1, \ldots, n$, denote the Bn iid samples from F_W obtained by calling $R_W(Bn)$. Replacing $Q(u_{b,i,d+1})$ by $W_{b,i}$ for $b = 1, \ldots, B$, $i = 1, \ldots, n$, is then equivalent to replacing the last coordinate of the n points in $\tilde{P}_{b,n}$ by independent U(0, 1) variates. With $W_{b,i} = Q(U_{b,i})$ where $U_{b,i} \overset{\text{ind.}}{\sim} \text{U}(0,1)$, $b = 1, \ldots, B$, $i = 1, \ldots, n$, $b = 1, \ldots, B$, we can write

$$\hat{\mu}_{b,n}^{\text{mc-rqmc}} = \frac{1}{n} \sum_{i=1}^{n} g(T_{\boldsymbol{Y}}(\boldsymbol{u}_{b,i,1:d}), Q(U_{b,i}))), \quad b = 1, \ldots, B. \tag{5}$$

From the inverse probability integral transform (see, e.g., [6, Theorem 2.1]), we know that $Q(U)$ for $U \sim \text{U}(0,1)$ and $R_n(1)$ have the same distribution, namely F_W. As such, unbiasedness of $\hat{\mu}_{b,n}^{\text{mc-rqmc}}$ (and therefore of $(1/B) \sum_{b=1}^{B} \hat{\mu}_{b,n}^{\text{mc-rqmc}}$) for μ follows immediately.

Note that only the first d coordinates of $\tilde{P}_{b,n}$ enter the estimation, so that the good projection properties of coordinate $d+1$ (and its interactions) are lost. Loosely speaking, the last coordinate of the point set we are effectively using to integrate the function g is unrelated with the first d. A better approach is to use the sampled $W_{b,i}$ to construct B empirical quantile functions $\hat{Q}_{n,b}$, $b = 1, \ldots, B$, and replace $Q(U_{b,i})$ by $\hat{Q}_{n,b}(u_{b,i,d+1}) = W_{b,(\lceil n u_{b,i,d+1} \rceil)}$, where, for $b = 1, \ldots, B$, we denote by $W_{b,(i)}$, $i = 1, \ldots, n$, the order statistics of $W_{b,1}, \ldots, W_{b,n}$, so $W_{b,(1)} \le \cdots \le W_{b,(n)}$. We define

$$\hat{\mu}_{b,n}^{\text{b-eqf}} = \frac{1}{n} \sum_{i=1}^{n} g(T_{\boldsymbol{Y}}(\boldsymbol{u}_{b,i,1:d}), W_{b,(\lceil n u_{b,i,d+1} \rceil)}), \quad b = 1, \ldots, B,$$

where superscript "b-eqf" indicates that in each randomization b, the empirical quantile function obtained in that randomization based on n samples is used (instead of Q). That is, in each of the B randomizations (each of which requires n function evaluations), estimate Q by its empirical quantile function $\hat{Q}_{n,b}$ obtained from n independent samples from F_W via a call to the black-box function $R_W(n)$. This procedure is different from the one in [10] for distributions supported on $[0,1)^s$: They do not sample using a black box NRVG to build an estimate of the empirical quantile function, rather, they use the original low-discrepancy sequence to count the relative number of elements below that value and use this number as the new point.

Note that as long as $\tilde{P}_{b,n,d+1} = \{u_{b,i,d+1} : i = 1, \ldots, n\}$ is *properly stratified*, i.e., has exactly one point in each interval of the form $[j/n, (j+1)/n)$ for $j \in \{0, \ldots, n-1\}$, each $W_{b,i}$, $i = 1, \ldots, n$ will be sampled exactly once when using $\tilde{P}_{b,n,d+1}$ to sample the empirical quantile function $\hat{Q}_{b,n}$. Hence an alternative way to describe the estimator $\hat{\mu}_{b,n}^{\text{b-eqf}}$ that is useful from an implementation perspective is

to realize that if the last coordinate of a given point $\boldsymbol{u}_{b,i}$ is the jth smallest value among those n last coordinates, we "stitch" $W_{b,(j)}$ to that ith point. Hence the last coordinate of $\tilde{P}_{b,n}$ is used to order the sample $W_{b,1}, \ldots, W_{b,n}$. Note that the problem of concatenating two samples within an RQMC-based approach also appears in the Array-RQMC method in [17, 20], where one needs to assign particles to points when paths are propagated. Also note that if $\tilde{P}_{b,n}$ is a digitally shifted or scrambled Sobol' point set with $n = b^k$ points or a randomly shifted rank-1 lattice, then $\tilde{P}_{b,n,d+1}$ is properly stratified; see [18].

The estimators $\hat{\mu}_{b,n}^{\text{b-eqf}}$ for $b = 1, \ldots, B$ are independent and as long as $\tilde{P}_{b,n,d+1}$ is properly stratified, they are also unbiased, see Proposition 1.

This alternative description gives rise to a slightly different estimator: Let $r^n(u_{b,i,d+1})$ be the rank of $u_{b,i,d+1}$ among $u_{b,1,d+1}, \ldots, u_{b,n,d+1}$. We then define the rank-based estimator as

$$\hat{\mu}_{b,n}^{\text{b-rk}} = \frac{1}{n} \sum_{i=1}^{n} g(T_{\boldsymbol{Y}}(\boldsymbol{u}_{b,i,1:d}), W_{b,(r^n(u_{b,i,d+1}))}), \quad b = 1, \ldots, B. \tag{6}$$

If $\tilde{P}_{b,n,d+1}$ is properly stratified, then $\hat{\mu}_{b,n}^{\text{b-rk}}$ and $\hat{\mu}_{b,n}^{\text{b-eqf}}$ coincide, and each sample $W_{b,i}$ is used exactly once. Otherwise, unlike $\hat{\mu}_{b,n}^{\text{b-eqf}}$, $\hat{\mu}_{b,n}^{\text{b-rk}}$ still uses every $W_{b,i}$ exactly once.

Proposition 1 *Let $b \in \{1, \ldots, B\}$ and let $\tilde{P}_{b,n,d+1}$ be properly stratified. Then $\hat{\mu}_{b,n}^{b\text{-}rk}$ (and therefore $\hat{\mu}_{b,n}^{b\text{-}eqf}$) is unbiased for μ.*

Proof Let $i \in \{1, \ldots, n\}$. We show that $\mathbb{E}(g(T_{\boldsymbol{Y}}(\boldsymbol{u}_{b,i,1:d}), W_{b,(r^n(u_{b,i,d+1})),n})) = \mu$. By definition, $(u_{b,i,1}, \ldots, u_{b,i,d+1}) \sim \mathrm{U}(0,1)^{d+1}$, in particular, $\boldsymbol{Y} := T_{\boldsymbol{Y}}(\boldsymbol{u}_{b,i,1:d}) \sim F_{\boldsymbol{Y}}$ is independent of $u_{b,i,d+1}$. Let $r^n(u_{b,i,d+1}) = K(i)$ (a random variable) and note that $(K(1), \ldots, K(n))$ is a permutation of $(1, \ldots, n)$ chosen according to some distribution (which may not be uniform because of the low-discrepancy properties of $\tilde{P}_{b,n}$). Then $W_{b,K(i)}$ is an element chosen from the list $W_{b,1}, \ldots, W_{b,n}$ according to some distribution, and the latter is an independent random sample from F_W. Hence, $W_{b,K(i)}$ and $\boldsymbol{Y}$ are independent, $(\boldsymbol{Y}, W_{b,K(i)}) \sim F_{\boldsymbol{Y}} \times F_W$ and the main claim follows by linearity of the expectation. □

The previous methods can be thought of as approximating the quantile function B times, each based on n samples obtained from the black box. In order to base our simulation on a sampling mechanism closer to inversion and thereby mimicking more closely the estimator in (4), we could instead construct a single rank-based quantile function estimator based on the Bn outputs $W_{b,i}, b = 1, \ldots, B, i = 1, \ldots, n$. That is, instead of reordering the n samples $W_{b,i}, i = 1, \ldots, n$ according to $u_{b,i,d+1}$ in each randomization $b = 1, \ldots, B$, separately, we reorder the Bn realizations $W_{b,i}$, $i = 1, \ldots, n$, $b = 1, \ldots, B$, according to the ranks of the $u_{b,i,d+1}$. That is, we construct the estimator

$$\hat{\mu}_{b,n}^{\text{1:B-rk}} = \frac{1}{n}\sum_{i=1}^{n} g(T_Y(\boldsymbol{u}_{b,i,1:d}), W_{(r^{Bn}(u_{b,i,1}))}), \quad b = 1, \ldots, B, \tag{7}$$

where $r^{Bn}(u_{b,i,d+1}) = k$ if $u_{b,i,d+1}$ is the kth smallest among the Bn uniforms $u_{1,1,d+1}, \ldots, u_{1,n,d+1}, \ldots, u_{B,1,d+1}, \ldots, u_{B,n,d+1}$.

We can replace the ranks by the empirical quantile function computed from $R_W(Bn)$, and obtain as an analog of $\hat{\mu}^{\text{b-eqf}}$ the estimator

$$\hat{\mu}_{b,n}^{\text{1:B-eqf}} = \frac{1}{n}\sum_{i=1}^{n} g(T_Y(\boldsymbol{u}_{b,i,1:d}), W_{(\lceil nBu_{b,i,1}\rceil)}), \quad b = 1, \ldots, B,$$

for a sample $W_1, \ldots, W_{nB} \overset{\text{ind.}}{\sim} F_W$ obtained by calling $R_W(Bn)$. The superscript "1:B-eqf" shall indicate that in all randomizations $1, \ldots, B$, the same quantile function estimator is used. Note that $\hat{\mu}_{b,n}^{\text{1:B-eqf}}$ are not independent anymore for $b = 1, \ldots, B$, the same applies to $\hat{\mu}_{b,n}^{\text{1:B-rk}}$.

Here we note that Pierre L'Ecuyer and his collaborators have proposed very efficient methods that combine conditional MC and RQMC to estimate quantile functions associated with a simulation output; see [25]. Our setting here is different, as we focus on estimating a univariate quantile function for the sole purpose of sampling.

2.2 Methods Based on a Generalized Pareto Approximation in the Tail

The methods presented in the previous section are purely nonparametric and amount to replacing the true quantile function Q by an empirical estimate thereof. Empirical quantile functions typically estimate quantiles away from the tail with reasonable accuracy; this does not hold for the tails if W is unbounded. However, approximating the tail of Q well is crucial for an effective RQMC procedure to outperform MC.

In the following, assume that W is supported on $[0, \infty)$ so that only the upper tail needs to be estimated. Since this is typically the case in practice, this is a rather weak assumption. If W is instead supported on $\mathbb{R}$, the methods described here can be applied to the positive and negative real line separately.

The main idea behind the methods presented in this section is the following: Given a random sample from F_W, estimate Q in the body (say, for $u \in (0, 0.9)$) by interpolation of the empirical quantile function and in the (right) tail based on a fitted generalized Pareto Distribution (GPD), which has a cumulative distribution function (cdf)

$$G_{\xi,\beta}(x) = \begin{cases} 1 - \left(1 + \frac{\xi x}{\beta}\right)^{-\frac{1}{\xi}}, & \text{if } \xi \neq 0, \\ 1 - \exp\left(-\frac{x}{\beta}\right), & \text{if } \xi = 0, \end{cases}$$

where $\beta > 0$ and the support is $x \in [0, \infty)$ when $\xi \geq 0$ and $x \in [0, -\beta/\xi]$ when $\xi < 0$.

Let F be any cdf and let $X \sim F$. Denote by $F_u(x) = \mathbb{P}(X - u \leq x \mid X > u)$ the excess distribution over the threshold u. Under weak assumptions, the Pickands–Balkema-de-Haan Theorem (see [7, Theorem 3.4.13]) implies that for large enough u one can approximate F_u by $G_{\xi,\beta}$.

In practice, ξ and β are estimated from given data. With estimates of ξ, μ at hand, we can compute $G^{-1}_{\xi,\beta}$ analytically, which, appropriately scaled, provides us with an estimate of F^{-1}. In what follows, assume F_W fulfills the assumptions underlying the Pickands–Balkema-de-Haan Theorem, and denote by $g_{\xi,\beta}$ the density of $G_{\xi,\beta}$. The following algorithm returns a quantile function estimator $\hat{Q}$ of Q.

Algorithm 1 Given $W_1, \ldots, W_{n'} \overset{\text{ind.}}{\sim} F_W$ and $\alpha \in (0, 1)$, construct an estimator $\hat{Q}$ for Q as follows:

1. Denote by $W_{(1)}, \ldots, W_{(n')}$ the order statistics of $W_1, \ldots, W_{n'}$.
2. Let $T = W_{(\lceil n'\alpha \rceil)}$ and denote by $N = |\{i \in \{1, \ldots, n'\} : W_i > T\}|$ the number of exceedances over T. Let $\tilde{W}_i = W_{(\lceil n'\alpha \rceil + i)} - T$ for $i = 1, \ldots, N$ be the excesses. Then maximize the log-likelihood function

$$l(\xi, \beta; \tilde{W}_1, \ldots, \tilde{W}_N) = \sum_{k=1}^{N} \log g_{\xi,\beta}(\tilde{W}_k)$$

   with respect to ξ and β numerically over their ranges to obtain the MLEs $\hat{\xi}$ and $\hat{\beta}$.
3. Return the function

$$\hat{Q}(u) = \begin{cases} (1 - \kappa) W_{(\lfloor (n'+1)u \rfloor)} + \kappa W_{(\lfloor (n'+1)u \rfloor + 1)}, & \text{if } u \leq \alpha, \\ T + \frac{\hat{\beta}}{\hat{\xi}} \left(\left(\frac{1-u}{1-\alpha} \right)^{-\xi} - 1 \right), & \text{otherwise,} \end{cases}$$

   where $\kappa = (n' + 1)u - \lfloor (n' + 1)u \rfloor$.

Algorithm 1 does not give any error estimates, nor do we have an a-priori guess of how large n should be. In order to obtain error estimates, one could use Algorithm 1 to obtain M independent estimators $\hat{Q}_m$, $m = 1, \ldots, M$, and estimate the error using a CLT argument. That is, the (absolute) error of $\hat{Q}(u) = (1/M) \sum_{m=1}^{M} \hat{Q}_m(u)$ for some fixed $u \in (0, 1)$ may be estimated via $3.5/\sqrt{M} \times \hat{\sigma}$, where $\hat{\sigma} = \mathrm{sd}(\hat{Q}_1(u), \ldots, \hat{Q}_M(u))$. If $\hat{Q}_m(u) - Q(u)$ follows approximately a $\mathbb{N}(0, \hat{\sigma}^2/M)$ distribution, we could be 99.95% confident that the error is within $\pm 3.5/\sqrt{M} \times \hat{\sigma}$; given the tails of the distribution are approximated, this assumption might be a bit optimistic.

With an error estimation procedure at hand, one can now construct the quantile function iteratively until a pre-specified error tolerance for the estimated absolute error is met. That is, one can specify knots $u'_1, \ldots, u'_N \in (0, 1)$ and error tolerances

$\varepsilon_1, \ldots, \varepsilon_N > 0$ and construct the quantile function with more and more points until the error tolerance at all knots is met. The choice of knots and error tolerances can be guided from the function g so that important subdomains have little error, or one can put most of the knots uniformly between 0 and 1 and the remaining ones in the tails. The main idea is summarized in the following algorithm.

Algorithm 2 Given $n_0 \in \mathbb{N}, \alpha \in (0, 1)$, NRVG R_W, knots $u'_1, \ldots, u'_N \in (0, 1)$, error tolerances $\varepsilon_1, \ldots, \varepsilon_N > 0$, maximum number of iterations $i_{\max}$, $B \in \mathbb{N}$, construct an estimator $\hat{Q}$ for Q as follows:

1. Set $i = 1$, and $S_b = \{\}$ for $b = 1, \ldots, B$.
2. Repeat
   a. For $b = 1, \ldots, B$,
      i. Set $S_b = S_b \cup \{R_W(n_0)\}$.
      ii. Call Algorithm 1 with input sample S_b to construct an estimated quantile function $\hat{Q}_b$.
   b. For $k = 1, \ldots, N$ set $e_k = 3.5/\sqrt{B} \times \mathrm{sd}(\hat{Q}_1(u'_k), \ldots, \hat{Q}_B(u'_k))$ as the estimated error at knot u'_k.
   c. Set $i = i + 1$.

   Until $e_k \le \varepsilon_k$ for $k = 1, \ldots, N$ or $i > i_{\max}$.
3. Return the estimated quantile function $\hat{Q}^{\text{eqf-gpd}}(u) = (1/B) \sum_{b=1}^{B} \hat{Q}_b(u)$.

The input argument $i_{\max}$ determines the maximum number of iterations allowed in case convergence cannot be achieved. Note that the superscript "eqf-gpd" shall indicate that the (interpolated) empirical quantile function is used in the body and a GPD approximation in the tail. For an implementation, in any iteration $i > 1$, results from the previous iterations should be reused; for instance, the MLE $(\hat{\xi}, \hat{\beta})$ from a previous iteration can be used as a starting value for the maximization of the log-likelihood function in the next iteration. In practice one could also return the $\hat{Q}_b$, $b = 1, \ldots, B$, so that for any $u \in (0, 1)$ one can compute $\hat{Q}(u)$ along with an error estimate.

Given an estimated quantile function, say $\hat{Q}^{\text{eqf-gpd}}$, an RQMC estimator for μ from (1) is given by

$$\hat{\mu}_{B,n}^{\text{eqf-gpd}} = \frac{1}{B} \sum_{b=1}^{B} \hat{\mu}_{b,n}^{\text{eqf-gpd}}, \tag{8}$$

where

$$\hat{\mu}_{b,n}^{\text{eqf-gpd}} = \frac{1}{n} \sum_{i=1}^{n} g(T_Y(\boldsymbol{u}_{b,i,1:d}), \hat{Q}^{\text{eqf-gpd}}(u_{b,i,d+1})), \quad b = 1, \ldots, B,$$

and the inputs $u_{b,i,d+1}$ and $\boldsymbol{u}_{b,i,1:d}$ are as in the previous section. In contrast to the estimators from Sect. 2.1, computing this estimator requires a two-stage procedure:

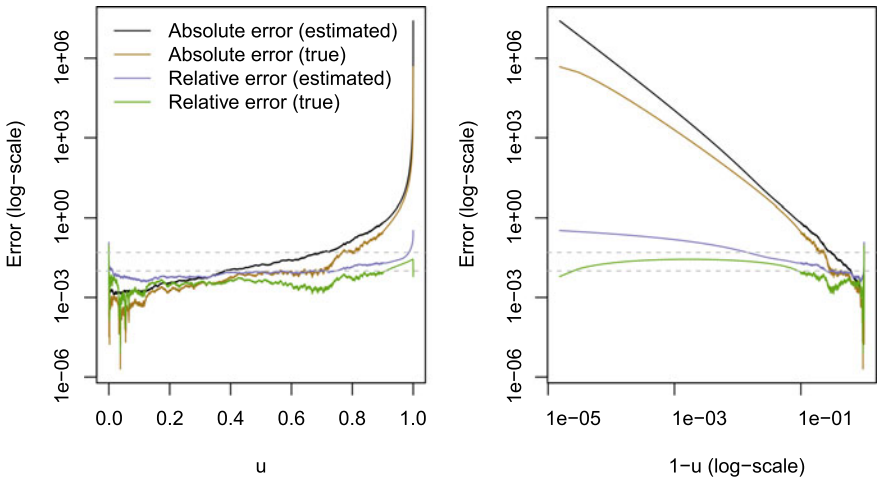


Fig. 1 Estimated and realized absolute and relative errors when estimating the quantile function of IG(1.2, 1.2) using Algorithm 2 with $n_0 = 7500$, $B = 20$

First, Algorithm 2 needs to be applied to compute the estimated quantile function $\hat{Q}^{\text{eqf-gpd}}$, which will then, in the second stage, be treated as the "true quantile function" when computing the estimator $\hat{\mu}^{\text{eqf-gpd}}$.

Example: Inverse-gamma. Consider $W \sim \text{IG}(1.2, 1)$. We use $n_0 = 7500$, $B = 20$, and uniform knots between 0.01 and 0.95 with relative error tolerance 0.025, one knot at 0.99 with relative error tolerance 0.075 and and another knot at 0.999 with relative error tolerance 0.1. The algorithm needed 10 iterations until convergence, so a total of 1 350 000 realizations of W. The approximation is very accurate and the true quantile lies within the approximated error bounds. This can be seen from Fig. 1, which displays realized and estimated absolute and relative errors.

Example: Expected shortfall of portfolio under a multivariate t distribution. The multivariate t distribution is a normal variance mixture distribution and falls into the general framework of this paper, if we assume that the quantile function of an inverse-gamma distribution is not available. We do this to compare our methods with the "best possible" estimator from (4). Let $\boldsymbol{\mu} \in \mathbb{R}^d$ and $\Sigma = AA^\top$ for some covariance matrix Σ. Then $\boldsymbol{X} \sim t_d(\nu, \boldsymbol{\mu}, \Sigma)$ has stochastic representation

$$\boldsymbol{X} = \boldsymbol{\mu} + \sqrt{W}\boldsymbol{Y}, \tag{9}$$

where $W \sim \text{IG}(\nu/2, \nu/2)$ independent of $\boldsymbol{Y} \sim \mathbb{N}_d(\mathbf{0}, \Sigma)$. For a continuous random variable $L \sim F$ with $\mathbb{E}(|L|) < \infty$ and level $\alpha \in (0, 1)$ small, expected shortfall is the mean conditional loss $\text{ES}_\alpha(L) = \mathbb{E}(L \mid L > F_L^{-1}(\alpha))$. In our simulation, we assume that $L = \mathbf{1}^\top \boldsymbol{X}$ where $\boldsymbol{X} \sim t_d(\nu, \mathbf{0}, \Sigma)$; it follows from the closedness of normal variance mixtures that $L \sim t_1(\nu, 0, \mathbf{1}^\top \Sigma \mathbf{1})$. The value of $\mu = \text{ES}_\alpha(L) := \mathbb{E}(g(\boldsymbol{Y}, W))$ is known in closed-form; see [23, Example 2.15]. This allows us to estimate the

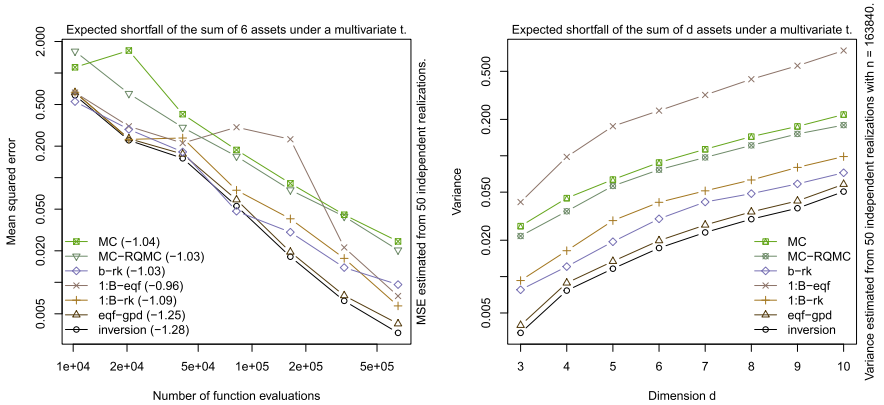


Fig. 2 Mean squared errors as a function of n (left) and variances as a function of d (right) when estimating $\mathrm{ES}_{0.95}(L)$ for $L = \mathbf{1}^\top \boldsymbol{X}$ where $\boldsymbol{X} \sim t_d(\nu, \mathbf{0}, \Sigma)$

mean squared error (MSE) and compare it with the variance. For a range of values of the total number of function evaluations, we report in Fig. 2 the mean squared error (MSE) and variance for various methods, each estimated by using $M = 50$ independent copies of the estimators, each of which is based on $B = 20$ repetitions. The numbers in brackets are the estimated regression coefficients; if the estimate is $\hat{\alpha}$, it means that the MSE as a function of the sample size n is in $\mathcal{O}(n^{-\alpha})$. Here and in what follows, we use a digitally shifted Sobol' sequence as implemented in the R package `qrng`; see [13]. Note that generating Sobol' points is faster than generating pseudo-random numbers using the Mersenne Twister, which is the default random number generator in R. All RQMC based estimators, including MC-RQMC from (5), outperform MC, though MC-RQMC gives only a moderate variance reduction. This is in contrast to b-rk, which for small n gives MSE similar to inversion, which we recall would not be available in a realistic setting where Q is unknown.

3 Combining AR with RQMC

Rather than working with a "black-box" RVG R_W, we assume in this section that W can be sampled using AR and explore how we can apply RQMC in this setting. Recall from (3) that we are interested in estimating $\mu = \mathbb{E}(g(\boldsymbol{Y}, W))$, so we need n samples $(\boldsymbol{Y}_i, W_i)$ where $W_i \sim F_W$. When using AR, there is no a-priori bound on how many uniforms are needed, so we have an a priori infinite-dimensional integration problem: If T_{AR} denotes the AR transformation, we can write $\mu = \mathbb{E}(h(\boldsymbol{U})) = \mathbb{E}(g(T_{\boldsymbol{Y}}(\boldsymbol{U}_{1:d}), T_{\mathrm{AR}}(\boldsymbol{U}_{(d+1):\infty}))$ with $\boldsymbol{U} \sim \mathrm{U}(0,1)^\infty$ and h appropriately defined. The integrand h is a non-monotone and discontinuous function of its input uniforms, a result from the acceptance decision. This can diminish the variance reduction effect of RQMC over MC.

i	Sample $\boldsymbol{Y}$				F^{-1}	AR
1	$u_{1,1}$	$u_{1,2}$	...	$u_{1,d}$	$u_{1,d+1}$	$u_{1,d+2}$
2	$u_{2,1}$	$u_{2,2}$	...	$u_{2,d}$	$u_{2,d+1}$	$u_{2,d+2}$
3	$u_{3,1}$	$u_{3,2}$	...	$u_{3,d}$	$u_{3,d+1}$	$u_{3,d+2}$
4	$u_{4,1}$	$u_{4,2}$	...	$u_{4,d}$	$u_{4,d+1}$	$u_{4,d+2}$
5	$u_{5,1}$	$u_{5,2}$	...	$u_{5,d}$	$u_{5,d+1}$	$u_{5,d+2}$
6	$u_{6,1}$	$u_{6,2}$	...	$u_{6,d}$	$u_{6,d+1}$	$u_{6,d+2}$
⋮	⋮				⋮	⋮

Fig. 3 Schematic description of AR-n. Gray coordinates in the same row correspond to rejected coordinates

i	Sample $\boldsymbol{Y}$			F^{-1}	AR	F^{-1}	AR	F^{-1}	AR	...
1	$u_{1,1}$	$u_{1,2}$ ...	$u_{1,d}$	$u_{1,d+1}$	$u_{1,d+2}$	$u_{1,d+3}$	$u_{1,d+4}$	$u_{1,d+5}$	$u_{1,d+6}$	
2	$u_{2,1}$	$u_{2,2}$ ...	$u_{2,d}$	$u_{2,d+1}$	$u_{2,d+2}$					
3	$u_{3,1}$	$u_{3,2}$ ...	$u_{3,d}$	$u_{3,d+1}$	$u_{3,d+2}$					
4	$u_{4,1}$	$u_{4,2}$ ...	$u_{4,d}$	$u_{4,d+1}$	$u_{4,d+2}$	$u_{4,d+3}$	$u_{4,d+4}$			

i	Sample $\boldsymbol{Y}$			F^{-1}	F^{-1}	F^{-1}	...	AR	AR	AR	...
1	$u_{1,1}$	$u_{1,2}$ ...	$u_{1,d}$	$u_{1,d+1}$				$u_{1,d+M+1}$			
2	$u_{2,1}$	$u_{2,2}$ ...	$u_{2,d}$	$u_{2,d+1}$	$u_{2,d+2}$	$u_{2,d+3}$		$u_{2,d+M+1}$	$u_{2,d+M+2}$	$u_{2,d+M+3}$	
3	$u_{3,1}$	$u_{3,2}$ ...	$u_{3,d}$	$u_{3,d+1}$				$u_{3,d+M+1}$			
4	$u_{4,1}$	$u_{4,2}$ ...	$u_{4,d}$	$u_{4,d+1}$	$u_{4,d+2}$			$u_{4,d+M+1}$	$u_{4,d+M+2}$		

Fig. 4 Schematic description of AR-d with consecutive (top) and blockwise (bottom) coordinate assignment. Gray coordinates in the same row correspond to rejected coordinates

We assume that W has density f_W over $(a, b) \subseteq \mathbb{R}$, we use the proposal density f having the same support (a, b) with quantile function F^{-1}, and that $c = \sup_{x\in(a,b)} f_W(x)/f(x) < \infty$.

A major difference between the application of RQMC and MC is that with the former, we need to carefully assign which coordinate of the points is used to sample which random variable, and there is typically more than one way to do so. As in the previous section, we assume that the first d coordinates $\boldsymbol{u}_{1:d}$ of $\boldsymbol{u} \in (0, 1)^\infty$ are used to sample from $F_{\boldsymbol{Y}}$. Algorithms 3 and 4 describe two AR methods to sample n copies of $(\boldsymbol{Y}, W)$; a schematic description is given in Figs. 3 and 4. The former method, henceforth referred to as AR-n, always uses coordinates $\{d + 1, d + 2\}$ in the AR part, and moves along the index i. If a point is rejected, just like the point in row $i = 1$ in Fig. 3, the algorithm tries again with point $i + 1$. That is, when sampling n points we move along the index of a randomized LDS with constant dimension $d + 2$. In contrast, Algorithm 4 (AR-d) samples the ith point by moving along the coordinates $\{d + 1, d + 2, d + 3, \dots\}$ of the ith point in the sequence until it is accepted; see the top of Fig. 4, where we assume that coordinates $d + 2j - 1$ and $d + 2j$ for $j = 1, 2, \dots$ are used for sampling from the proposal and sampling from the AR decision, respectively. Another possibility to assign the coordinates for the AR part is to consider two blocks of size M (chosen so that, with high probability, M

trials are sufficient to accept a point), where the coordinates in the first block are used for the sampling in Step 2(a)i and the coordinates in the second block determine the acceptance decision in Step 2(a)ii. This version of AR-d is illustrated at the bottom of Fig. 4.

Algorithm 3 (AR-n) Let $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \} \subset (0,1)^{d+2}$ be a randomized LDS. Sample n copies of $(\boldsymbol{Y}, W)$ as follows.

1. Set $j = 1$, $O_n = \{\}$.
2. For $i = 1, \ldots, n$,
   a. Repeat
      i. Compute $W = F^{-1}(u_{j,d+1})$ and set $U = u_{j,d+2}$.
      ii. If $U > f_W(W)/(cf(W))$ set $j = j + 1$ Else
         Set $O_n = O_n \cup \{(T_{\boldsymbol{Y}}(\boldsymbol{u}_{j,1:d}, W)\}$
         Set $j = j + 1$ and break;
3. Return O_n.

The main difference between AR-n and AR-d is that in the former approach, points in the sequence are skipped, and, effectively, a subset of size n of the first $N > n$ points in the sequence is used to integrate g, whereas in AR-d we always use the first n points in the sequence and move along the coordinates.

Algorithm 4 (AR-d) Let $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_n\} \subset (0,1)^{\infty}$ be a randomized low discrepancy point-set. Sample n copies of $(\boldsymbol{Y}, W)$ as follows.

1. Set $O_n = \{\}$.
2. For $i = 1, \ldots, n$,
   a. For $j = 1, 2, \ldots,$
      i. Compute $W = F^{-1}(u_{i,d+2j-1})$.
      ii. If $u_{i,d+2j} \le f_W(W)/(cf(W))$:
         A. Set $O_n = O_n \cup \{(T_{\boldsymbol{Y}}(\boldsymbol{u}_{i,1:d}, W)\}$
         B. Break.
3. Return O_n.

A potential advantage of AR-d over AR-n for numerical integration is that it really only uses the first n points of the LDS rather than a subset of the first $N > n$ points in the sequence. In order to highlight this point, assume that our integrand does not depend on W and that $n = 2^k$. When estimating μ based on AR-d, we will then use the first 2^k points of the underlying LDS and keep all its good projection properties. In contrast, using AR-n, we only use a subset of size 2^k of the first $N > 2^k$ points, thereby potentially loosing some of the good projection properties of the LDS. This point is illustrated in Fig. 5, where we first sample (W_i, Y_{i1}, Y_{i2}) where $Y_{ij} \sim \mathbb{N}(0,1)$, $j = 1, 2$, and $W_i \sim \Gamma(1.2, 1)$ for $i = 1, \ldots, n = 2^7$, and then set $\boldsymbol{U}_i = (F_W(W_i), \Phi(Y_{i1}), \Phi(Y_{i2}))$ for $i = 1, \ldots, n$. By the probability integral transformation, $\boldsymbol{U}_i \sim \mathrm{U}(0,1)^3$. Note that if we had used inversion to sample the

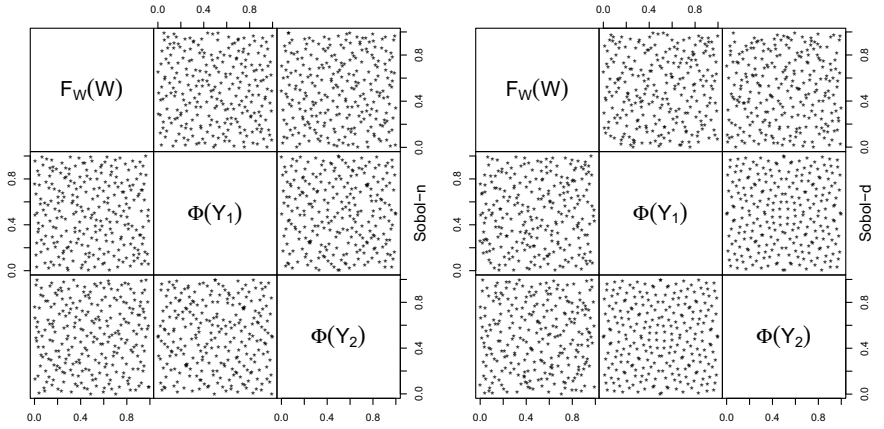


Fig. 5 Pairs plot of $(F_W(W_i), \Phi(Z_{i1}), \Phi(Z_{i2})) \sim \mathrm{U}(0,1)^3$, where the trivariate points were sampled with AR-n (left) and with AR-d (right) for $W \sim \Gamma(1.2, 1)$ and $n = 2^8$

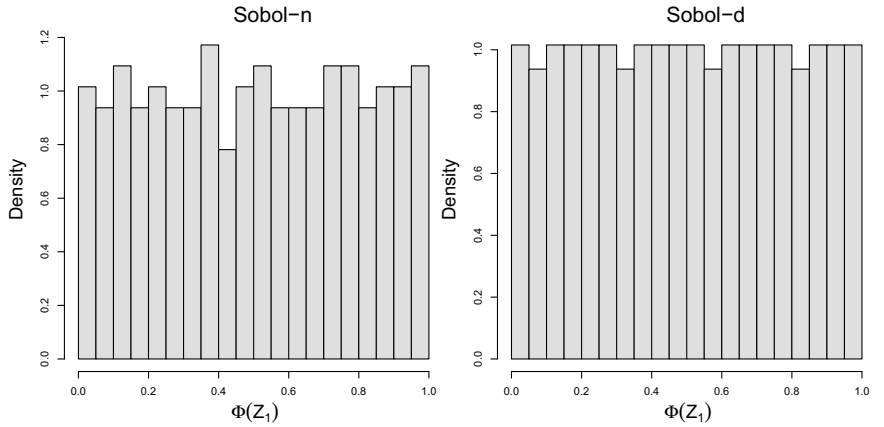


Fig. 6 Histogram of U_1 when constructed with AR-n (left) and AR-d (right)

W_i, the points would be exactly the original LDS. Note how Sobol'-d gives a point set with better marginal uniformity than Sobol'-n, which is also confirmed in the histogram of the first standardized coordinate in Fig. 6. Note that if we had 2^k bins with $k \leq 7$ we would see a flat histogram on the right-hand side of this figure; here and it what follows, we use the AR samplers for the Gamma distribution from [2, 15] for $\nu > 1$ and $\nu < 1$, respectively.

Next, we show in Propositions 2 and 3 that both algorithms produce point sets with the correct distribution.

Proposition 2 *Each $\boldsymbol{x} \in O_n$ produced by Algorithm 3 has distribution $F_{\boldsymbol{Y}} \times F_W$.*

Proof It suffices to show that the two numbers used to sample from the proposal and the acceptance decision are independent U(0, 1) random variables. The rest follows from the correctness of the AR algorithm; see, e.g., [8] for a proof. Let $\boldsymbol{x} = (\boldsymbol{Y}, W) \in O_n$. Then there is a $j \in \{1, 2, \dots\}$ such that $W = F^{-1}(U_1)$ and $U_2 \le f_W(W)/(cf(W))$ where $U_1 = u_{j,d+1}$ and $U_2 = u_{j,d+2}$ satisfy $U_1, U_2 \overset{\text{ind.}}{\sim} \mathrm{U}(0,1)$ by the randomization of the LDS. □

Proposition 3 *Each $\boldsymbol{x} \in O_n$ produced by Algorithm 4 has distribution $F_{\boldsymbol{Y}} \times F_W$.*

Proof Since we assumed that the chosen LDS is randomized so that each $\boldsymbol{u}_{b,i} \sim U(0,1)^{d+1}$, the coordinates $u_{i,d+j}$ used in Step 2(a)i and 2(a)ii are independent U(0, 1) for $j \ge 1$. The claim follows from the correctness of the AR algorithm. □

Our investigation of AR-d was motivated by the argument that AR corresponds to infinite-dimensional integration; see [9, pp. 62–63], who also notes that "potential drawback of AR methods, compared with the inverse transform method, is that their outputs are generally neither continuous nor monotone functions of the input uniforms." We can address the monotonicity by using the rank transformations from the black box setting in Sect. 2: that is, we re-order the outputs $W_1, \dots, W_n$ so that their order matches the ordering of $u_{1,d+1}, \dots, u_{n,d+1}$. If $n = 2^k$, this is exactly the b-rk method from Sect. 2 applied with the output of AR-d as a "black box". Note that this makes the AR-d output monotone in coordinate $d + 1$ of the underlying LDS. Note that with AR-n, we always use $u_{i,d+1}$ for some i to sample from the envelope via inversion, so that the monotonicity in this coordinate is already given.

Expected shortfall example continued. We perform the same example as in Sect. 2.2, but this time, using the AR based methods instead of the black-box setting. See Fig. 7. All AR based methods outperform pure MC and MC-RQMC, and the

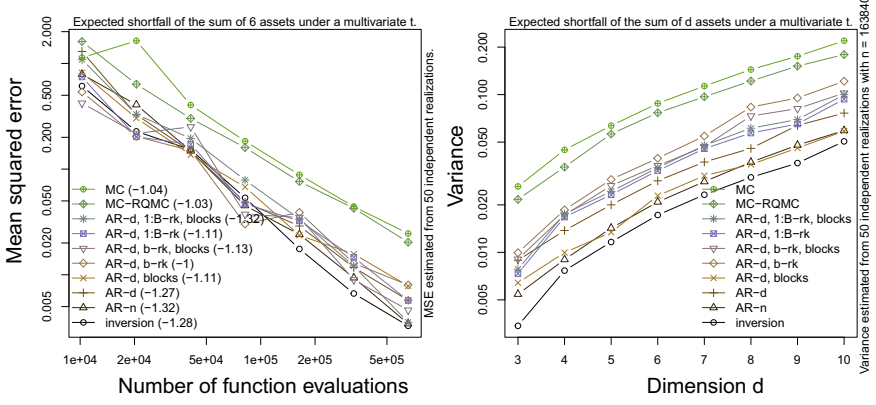


Fig. 7 Mean squared errors as a function of n (left) and variances as a function of d (right) when estimating $\mathrm{ES}_{0.95}(L)$ for $L = \mathbf{1}^\top \boldsymbol{X}$ where $\boldsymbol{X} \sim t_d(\nu, \mathbf{0}, \Sigma)$

convergence speed of AR-n and AR-d, 1:B-rk are almost as high as for the method "inversion", which we recall would not be available in a realistic setting.

4 Application: Basket Option Pricing

Consider the problem of estimating the value of a Basket call option with strike K, whose payoff with maturity $T = 1$, can be expressed as

$$\mu_{\text{bskt}} = e^{-r}\mathbb{E}\left(\max\left\{\frac{1}{d}\sum_{j=1}^{d} S_j - K, 0\right\}\right);$$

we assume that the dependence of the log-normal assets S_j, $j = 1, \ldots, d$, is modeled via a t-copula. As such, the assets S_j have stochastic representation

$$S_j = F_{\text{LN}}^{-1}(U_j), \quad U_j = F_{t_\nu}(X_j), \quad j = 1, \ldots, d, \quad \boldsymbol{X} \sim t_d(\nu, \mathbf{0}, \Sigma);$$

here, Σ is a correlation matrix. The t copula is one of the most widely used copulas in risk management; see, e.g., [4] for more. Pricing basket options is a popular problem to perform RQMC experiments; see, e.g., [16]. The value of μ_{bskt} is not known, so we look at the estimated variances for the following methods:

- `MC`: Use MC for W and $\boldsymbol{Y}$;
- `MC-RQMC`: use MC for W and RQMC for $\boldsymbol{Y}$, i.e., compute $\hat{\mu}^{\text{mc-rqmc}}$ in (5).
- `AR-d`: Use Algorithm 4, i.e., sample W based on AR whilst moving along the coordinates of a point in the LDS until acceptance.
- `AR-n`: Use Algorithm 3, i.e., sample W based on AR whilst moving along the index of the point in the LDS until acceptance.
- `AR-d, b-rk`: Use AR-d in each repetition b and additionally reorder the n samples $W_{1,b}, \ldots, W_{n,b}$ according to $u_{1,b,d+1}, \ldots, u_{n,b,d+1}$ for $b = 1, \ldots, B$.
- `AR, 1:B-rk`: Use AR-d and sort all the sample $W_{1,1}, \ldots, W_{n,B}$ according to $u_{1,1,d+1}, \ldots, u_{n,B,d+1}$.
- `b-rk`: Treat R_W as black-box and compute $\hat{\mu}_{b,n}^{\text{b-rk}}$ from (6) for $b = 1, \ldots, B$.
- `1:B-rk`: Treat R_W as black-box and compute $\hat{\mu}_{b,n}^{\text{1:B-rk}}$ from (7) for $b = 1, \ldots, B$.
- `eqf-gpd`: First, build gpd based estimate $\hat{Q}$ using samples obtained from the black box R_W, then treat it as true Q and proceed with inversion; see$\hat{\mu}^{\text{eqf-gpd}}$ in (8).
- `inversion`: Compute the inversion based estimator $\hat{\mu}_{b,n}^{\text{rqmc}}$ from (4) for $b = 1, \ldots, B$ using the true quantile function.

The last method "inversion" is not available in a realistic setting like the stable example at the end of this section, but is included here to compare our methods with the best possible one. All methods (except for MC) sample the multivariate normal random vector $\boldsymbol{Y}$ based on inversion of a digitally shifted Sobol' sequence.

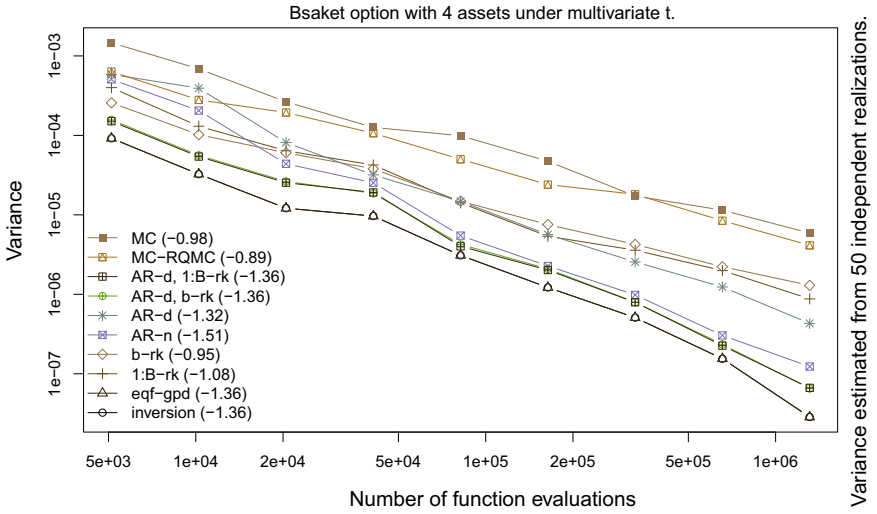


Fig. 8 Variances when estimating μ_{bskt} under a t copula with $\nu = 2.2$ dof, $r = 0.01$, $\sigma = 0.2$ (volatility for all stocks) as a function of n

The results in Fig. 8 indicate that using RQMC for sampling $\boldsymbol{Y}$ gives at least a modest variance reduction. Furthermore, treating the sampler as a black box and reordering the W samples as described in Sect. 2 gives further variance reduction. On the right hand side we see the AR methods from Sect. 3 which all give lower variance than the black-box methods; this makes sense as we are directly manipulating the sampler with some rank reordering, outperform AR-n.

Next, we alter this example so that we end up with a model where the quantile function of W is not as easily available as the quantile function of the inverse-gamma distribution (via `qgamma()`). To this end, we replace the t copula with a GIG-mixture copula. A random vector $\boldsymbol{X}$ has a GIG-mixture distribution if it follows the stochastic representation (9) with $W \sim \text{GIG}(\beta, \lambda)$; see [14] for a definition and an AR algorithm.

The marginal distribution functions F_j of X_j needed to compute the copula sample are not known, so we denote by $\hat{F}_j(x) = (n+1)^{-1} \sum_{i=1}^{n} \mathbf{1}_{\{X_{ij} \le x\}}$ the empirical distribution function of X_j, and instead compute the pseudo observations $\boldsymbol{U}_i = (\hat{F}_1(X_{i1}), \ldots, \hat{F}_d(X_{id}))$ for $i = 1, \ldots, n$.

In this example, we also include the method `runuran`, implemented in the R package with the same name [22]. By using the density function as input, it approximates the distribution function numerically and builds an approximation of the quantile function using splines; see [5]. It was demonstrated in [21] that this method works well for the GIG distribution.

Figure 9 shows estimated variances as a function of n (left) and the number of assets d (right); the lines for the methods runuran, eqf-gpd, 1:B-rk and b-rk are overlapping. These are the best methods in terms of estimated variance. All RQMC

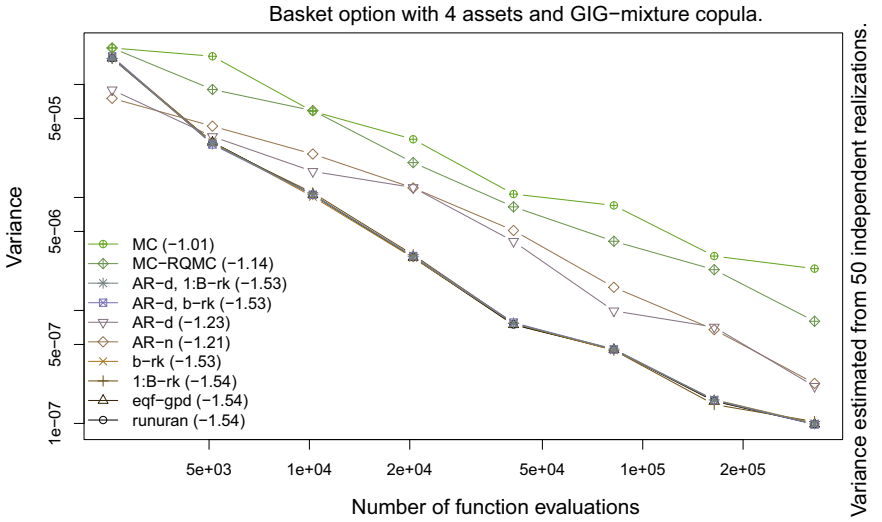


Fig. 9 Variances when estimating μ_{bskt} under a GIG mixture copula with $\lambda = 0.5$, $\beta = 0.3$, $r = 0.01$, $\sigma = 0.2$ (volatility for all stocks) as a function of n

Table 1 Average run times in seconds (top) and estimated efficiencies (bottom) when computing various estimators with sample size $n = 20 \times 2^{12}$ to estimate μ_{bskt} under a 9-dimensional GIG mixture copula with $\beta = 0.3$ and $\lambda = 0.5$

	Runuran	eqf-gpd	1:B-rk	b-rk	AR-n	AR-d	AR-d, b-rk	AR-d, 1:B-rk	MC-RQMC	MC
CPU	0.342	3.965	0.545	0.545	0.541	0.530	0.532	0.553	0.548	0.542
REff	7.32	0.63	4.60	4.62	2.70	2.54	4.72	4.52	1.61	1.00

based methods outperform MC and MC-RQMC gives the smallest variance reductions while AR-d and AR-n yield a modest variance reduction.

In the first row of Table 1, we show average run-times (in seconds) when computing various estimators when $n = 20 \times 2^{12}$ and $d = 9$. All methods, with the exception of eqf-gpd and runuran, take roughly the same time. Recall that with eqf-gpd, the idea is to estimate the quantile function Q once only using R_W, and then use it as a true quantile function for all subsequent simulations.The runuran method is the fastest. The second row of Table 1 shows relative efficiencies. The efficiency of a method is defined by $(\text{CPU} \cdot \text{Var})^{-1}$ and we prefer methods with large efficiency. We estimate these efficiencies and standardize them by the efficiency of pure MC. Method "eqf-gpd", due to its long run time, is the least efficient method, while the runuran method is most efficient, though we remark that it is the only method displayed that had access to the density function of W. The black box methods 1:B-rk and b-rk substantially outperform MC-RQMC, giving support for this simple re-ordering scheme. The re-ordering also helps the AR-based methods.

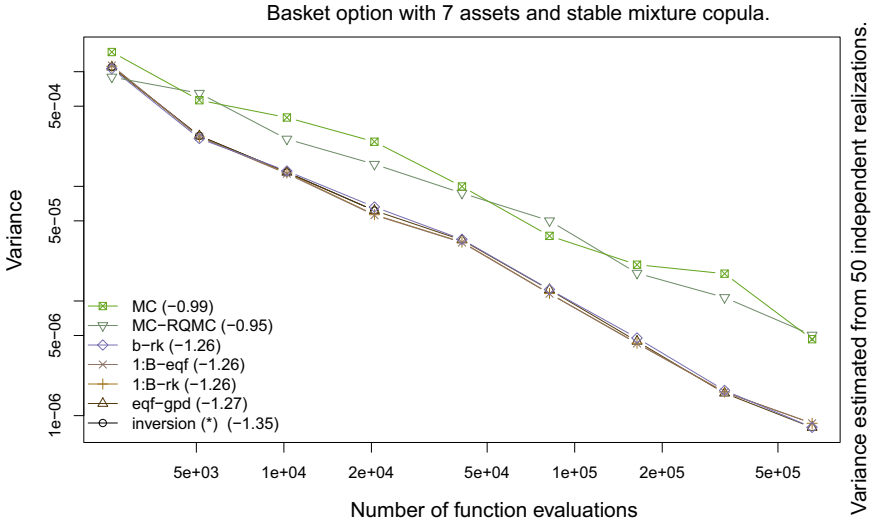


Fig. 10 Variances when estimating μ_{bskt} under a stable mixture copula with $\alpha = 0.6$, $\beta = 1$ and $\gamma = \cos((\pi/2)\alpha)^{1/\alpha}$, $r = 0.01$, $\sigma = 0.2$ (volatility for all stocks) as a function of n. (*) The experiment for "inversion" was only performed up to $n = 2 \times 10^4$, so the regression coefficient was computed using a smaller sample than the other coefficients

Table 2 Average run times in seconds (top) and estimated efficiencies (bottom) when computing various estimators with total sample size $n = 20 \times 2^{10}$ to estimate μ_{bskt} under a 7-dimensional stable mixture copula with $\alpha = 0.9$, $\beta = 1$ and $\gamma = 1$

	Inversion	eqf-gpd	1:B-rk	b-rk	MC-RQMC	MC
CPU	23.8	4.4	0.4	0.4	0.4	0.4
REff	0.06	0.33	3.64	3.59	1.26	1.00

Finally, we repeat the same experiment with the only change being that now we assume that W follows a stable distribution; see [1] for a sampling algorithm for the stable distribution. Note that not even the density of a stable distribution can be easily computed, hindering the application of numerical integration schemes to approximate the quantile function. In our simulation, we use the R package `stabledist`; see [3]. We use the function `rstable()` as a "black box" NRVG and the function `qstable()` to compare against the inversion method; note that it relies on numerical integration. The results are displayed in Fig. 10, where we used the parameters $\alpha = 0.6$, $\beta = 1$ and $\gamma = \cos((\pi/2)\alpha)^{1/\alpha}$ for the stable distribution so that the support is $[0, \infty)$. Due to numerical problems with `qstable()` it was only used for sample sizes up to 2×10^4. See Table 2 for run times: For the chosen sample size, even our eqf-gpd method is faster than inversion. Our rank based methods are the most efficient methods in this example.

5 Conclusion

We explored the question how RQMC can be applied to estimate $\mu = \mathbb{E}(g(\boldsymbol{Y}, W))$ when all components but one can be sampled via inversion, and the remaining one W by calling a NRVG only. Our proposed algorithms in the black box setting were motivated by the fact that RQMC works best when combined with inversion, so that our methods aim at mimicking this observation by exploiting the sample to estimate the quantile function. In Sect. 3, we assumed the existence of an AR algorithm, and motivated an AR-d algorithm that samples along the coordinates rather than moving along the sequence. Our numerical results indicate that RQMC can still provide a substantial variance reduction when combined with a NRVG. In particular we saw that the re-ordering methods outperform MC-RQMC (where we merely combine RQMC with pseudo-random sampling of W). Furthermore, we saw that moving along the coordinates as we do in AR-d can give better results than the previously proposed AR-n methods. With the methods in this paper at hand, we could extend the algorithms in [11, 12] to estimate various quantities related to multivariate normal variance mixture distributions, such as the distribution function. Furthermore, we plan to address some questions of computational nature, such as exploring efficient implementations of AR-d based on point sets that are easily extensible in the number of coordinates, such as Korobov rules based on well-chosen generators a; see [18]. Finally, this paper mostly focused on numerical comparisons of different RQMC-based algorithms based on digitally shifted Sobol' sequences. In the near future we plan to study settings under which it might be possible to obtain theoretical results demonstrating the superiority of our proposed RQMC-based methods (perhaps based on scramblings rather than shifts) over MC.

Acknowledgements We thank the reviewers for their comments, which helped us improve this chapter. The second and third authors are grateful for the financial support of NSERC via grant RGP 238959 and RGPIN-2020-04897, respectively.

References

1. Chambers, J., Mallows, C., Stuck, B.: A method for simulating stable random variables. J. Amer. Stat. Assoc. **71**(354), 340–344 (1976). https://doi.org/10.1080/01621459.1976.10480344
2. Cheng, R.: The generation of Gamma variables with non-integral shape parameter. J. Roy. Stat. Soc.: Ser. C (Appl. Stat.) **26**(1), 71–75 (1977)
3. Wuertz, D., Maechler, M., Rmetrics core team members: Stabledist: Stable Distribution Functions (2016). https://CRAN.R-project.org/package=stabledist. R package version 0.7-1
4. Demarta, S., McNeil, A.: The t copula and related copulas. Int. Stat. Rev. **73**(1), 111–129 (2005). https://doi.org/10.1111/j.1751-5823.2005.tb00254.x
5. Derflinger, G., Hörmann, W., Leydold, J.: Random variate generation by numerical inversion when only the density is known. ACM Trans. Model. Comput. Simul. (TOMACS) **20**(4), 1–25 (2010)
6. Devroye, L.: Non-Uniform Random Variate Generation. Springer, New York (1986). https://doi.org/10.1007/978-1-4613-8643-8

7. Embrechts, P., Klüppelberg, C., Mikosch, T.: Modelling extremal events. Br. Actuar. J. **5**(2), 465–465 (1999)
8. Flury, B.: Acceptance-rejection sampling made easy. SIAM Rev. **32**(3), 474–476 (1990)
9. Glasserman, P.: Monte Carlo Methods in Financial Engineering, vol. 53. Springer Science & Business Media, Berlin (2013)
10. Hartinger, J., Kainhofer, R.: Non-uniform low-discrepancy sequence generation and integration of singular integrands. In: Niederreiter, H., Talay, D. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2004, pp. 163–179. Springer, Berlin (2006)
11. Hintz, E., Hofert, M., Lemieux, C.: Grouped normal variance mixtures. Risks **8**(4), 103 (2020). https://doi.org/10.3390/risks8040103
12. Hintz, E., Hofert, M., Lemieux, C.: Normal variance mixtures: distribution, density and parameter estimation. Comput. Stat. Data Anal. **157C**, 107175 (2021). https://doi.org/10.1016/j.csda.2021.107175
13. Hofert, M., Lemieux, C.: qrng: (Randomized) Quasi-Random Number Generators (2019). https://CRAN.R-project.org/package=qrng. R package version 0.0-7
14. Hörmann, W., Leydold, J.: Generating generalized inverse Gaussian random variates. Stat. Comput. **24**(4), 547–557 (2014). https://doi.org/10.1007/s11222-013-9387-3
15. Kundu, D., Gupta, R.: A convenient way of generating Gamma random variables using generalized exponential distribution. Comput. Stat. Data Anal. **51**(6), 2796–2802 (2007). https://doi.org/10.1016/j.csda.2006.09.037
16. L'Ecuyer, P.: Quasi-Monte Carlo methods in finance. In: Proceedings of the 2004 Winter Simulation Conference, vol. 2, pp. 1645–1655. IEEE (2004)
17. L'Ecuyer, P., Lécot, C., Tuffin, B.: A randomized quasi-Monte Carlo simulation method for Markov chains. Oper. Res. **56**(4), 958–975 (2008)
18. L'Ecuyer, P., Lemieux, C.: Variance reduction via lattice rules. Manage. Sci. **46**(9), 1214–1235 (2000)
19. L'Ecuyer, P., Lemieux, C.: Recent advances in randomized quasi-Monte Carlo methods. In: Dror, M., L'Ecuyer, P., Szidarovszki, F. (eds.) Modeling Uncertainty: An Examination of Stochastic Theory, Methods, and Applications, pp. 419–474. Kluwer Academic Publishers, Boston (2002)
20. L'Ecuyer, P., Munger, D., Lécot, C., Tuffin, B.: Sorting methods and convergence rates for array-RQMC: some empirical comparisons. Math. Comput. Simul. **143**, 191–201 (2018)
21. Leydold, J., Hörmann, W.: Generating generalized inverse Gaussian random variates by fast inversion. Comput. Stat. & Data Anal. **55**(1), 213–217 (2011)
22. Leydold, J., Hörmann, W.: Runuran: R Interface to the 'UNU.RAN' Random Variate Generators (2020). https://CRAN.R-project.org/package=Runuran. R package version 0.30
23. McNeil, A., Frey, R., Embrechts, P.: Quantitative Risk Management: Concepts. Techniques and Tools. Princeton University Press (2015). https://doi.org/10.1007/s10687-017-0286-4
24. Moskowitz, B., Caflisch, R.: Smoothness and dimension reduction in quasi-Monte Carlo methods. Math. Comput. Model. **23**(8–9), 37–54 (1996). https://doi.org/10.1016/0895-7177(96)00038-6
25. Nakayama, M., Kaplan, Z.T., L'Ecuyer, P., Tuffin, B.: Quantile estimation via a combination of conditional Monte Carlo and randomized quasi-Monte Carlo. In: Proceedings of the 2020 Winter Simulation Conference (2020)
26. Nguyen, N., Ökten, G.: The acceptance-rejection method for low-discrepancy sequences. Monte Carlo Methods Appl. **22**(2), 133–148 (2016). https://doi.org/10.1515/mcma-2016-0104
27. Rosenblatt, M.: Remarks on a multivariate transformation. Ann. Math. Stat. **23**(3), 470–472 (1952). https://doi.org/10.1214/aoms/1177729394
28. Wang, X.: Improving the rejection sampling method in quasi-Monte Carlo methods. J. Comput. Appl. Math. **114**(2), 231–246 (2000). https://doi.org/10.1016/S0377-0427(99)00194-6
29. Zhu, H., Dick, J.: Discrepancy bounds for deterministic acceptance-rejection samplers. Electr. J. Stat. **8**(1), 678–707 (2014). https://doi.org/10.1214/14-EJS898

A Generalized Transformed Density Rejection Algorithm

Wolfgang Hörmann and Josef Leydold

Abstract Transformed density rejection is a very flexible method for generating non-uniform random variates. It is based on the acceptance-rejection principle and utilizes a strictly monotone map that transforms the given density into a concave or convex function. Hat function and squeezes are then constructed by means of tangents and secant. We present a new method that works for arbitrary one time continuously differentiable densities. It requires together with the log-density and its derivative a partition of the domain into subdomains that contain at most one inflection point. This improves a previous method of the authors in which also the second derivative is required. We show how the algorithm can be applied to generate from the Generalized Inverse Gaussian distribution, from the Generalized Hyperbolic distribution and from the Watson distribution. The new algorithm can also generate random variates from truncated distributions without problems.

Keywords Non-uniform random variate generation · Black-box algorithm · Transformed density rejection · Adaptive rejection sampling

1 Introduction

Monte Carlo methods and stochastic simulation are very powerful tools for computing metric values in models. A crucial step is the sampling of uniform random numbers and non-uniform random variates. For the latter acceptance-rejection sampling is often used. Then an upper bound h (called *hat* function) and optionally a lower bound s (called *squeeze*) has to be found that satisfy $0 \le s(x) \le f(x) \le h(x)$. Hat function h must be some multiple of a density function that allows for easy sampling

W. Hörmann
Department of Industrial Engineering, Boğaziçi University, 34342 Bebek-İstanbul, Turkey
e-mail: hormannw@boun.edu.tr

J. Leydold (✉)
Institute for Statistics and Mathematics, Vienna University of Economics and Business, Welthandelsplatz 1, Building D4, 1020 Vienna, Austria
e-mail: josef.leydold@wu.ac.at

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_14

from (typically by inversion) and squeeze s may be used to reduce the computational expense of evaluating f. Once values of h and s have been found, to generate a value of X from a distribution with density f, the following steps are necessary:

1. Generate a random variate X with density proportional to h.
2. Generate a (0, 1) uniform random number, U.
3. If $U\,h(X) \le s(X)$, then return X.
4. If $U\,h(X) \le f(X)$, then return X.
5. Otherwise, try again.

Although executing the five steps above is simple, the challenge in implementing acceptance-rejection sampling is in finding appropriate functions h and s. There exist many papers proposing such functions especially tailored for standard distributions, see, e.g., [9] for an extensive survey.

Devroye [8] discussed a different approach and proposed a general method to construct hat functions that works for all distributions with log-concave densities. Notice here that the given density need not be normalized. That is, any multiple of a density (with unknown proportionality factor) can be used. Gilks and Wild [11] use tangents and secants of the log-density to construct hat and squeeze functions, resp. Thus the hat distribution is a mixture of truncated exponentially distributed random variates with disjoint domains. Hence sampling by inversion is fast and simple. The interval boundaries are computed as the intersection points of the tangents. The initial subdivision is then refined by adaptive rejection sampling (ARS).

Hörmann [12] generalized this idea for the class of T-concave distributions. A density f is called T-concave if the transformed density $\tilde{f} = T \circ f$ is concave, where $T\colon (0, \infty) \to \mathbb{R}$ is a differentiable and monotonically increasing transformation. If f is T-concave, the tangent $\tilde{t}(x) = \alpha + \beta x$ to $\tilde{f}$ is greater than $\tilde{f}$ for all x in the domain of f, making the function $t(x) = T^{-1}\left[\tilde{t}(x)\right] = T^{-1}(\alpha + \beta x)$ a hat function to f. He also suggested a class of Box-Cox-like transformations where again sampling from the hat distribution by inversion is quite cheap, see Table 1. We want to note here that a T_c-concave density is also T_{c_1}-concave for every $c_1 \le c$. Furthermore, we need $c > -1$ for unbounded intervals as otherwise the integral of the hat function is unbounded. For a detailed discussion we refer to [13].

Table 1 The family T_c of transformations. F_T denotes the antiderivative of T_c^{-1}

c	$T_c(x)$	$T_c^{-1}(x)$	$F_T(x)$	$F_T^{-1}(x)$
> 0	x^c	$x^{1/c}$	$\frac{c}{c+1}x^{(c+1)/c}$	$\left(\frac{c+1}{c}x\right)^{c/(c+1)}$
0	$\log(x)$	e^x	e^x	$\log(x)$
< 0	$-x^c$	$(-x)^{1/c}$	$-\frac{c}{c+1}(-x)^{(c+1)/c}$	$-\left(-\frac{c+1}{c}x\right)^{c/(c+1)}$
$-1/2$	$-1/\sqrt{x}$	$1/x^2$	$-1/x$	$-1/x$
-1	$-1/x$	$-1/x$	$-\log(-x)$	$-\exp(-x)$

Similarly, if f is T-concave, the secant to the transformed density, $\tilde{r}$, can be used to construct the squeeze function, s, for the density in a given interval. Evans and Swartz [10] show that the opposite applies (in that tangents are used to construct the squeeze function and secants are used to construct the hat function) when f is T-convex.

It is quite obvious that this approach works for an arbitrary distribution with differentiable density when we partition its domain into intervals where the density is either T-concave or T-convex. For unbounded sub-intervals it is required that the density is T-concave. Although it is not necessary to use the same transformation in each subdomain, identifying these intervals requires the inflection points of $\tilde{f}$, which are often difficult to obtain.

For that reason Botts [5] relaxed the requirement of knowing the exact position of the inflection points. He proposes a method requiring a subdivision into intervals where the transformed density is either concave, convex, or has exactly one inflection point. For the latter case, he introduces an additional transformation and compiles a new sampling algorithm.

This idea has been simplified by [6]. It avoids the necessity of an additional transformation and relaxes the rather strong properties of the densities. Instead the user has to provide an implementation of the second derivative of the (log-) density. A ready-to-use version of the proposed algorithm is provided as **R** package `Tinflex`, see [14].

A closer look at the `Tinflex` algorithm reveals that it is enough to know the *sign* of the second derivative. In this contribution we thus show how to develop a method that only requires the transformed density and its derivative for constructing hat and squeeze in a reliable way at the expense of a slightly increased complexity of the setup part of the generator. We are convinced that from a mathematical point of view it is clear that an algorithm with less input is more elegant and preferable. Also we can assume that most users will be glad to have to supply only two functions instead of three. It is certainly true that numeric or automatic derivation could be used to reach that aim of more convenience for the user. But of course that depends also on the experience the user has with software for numeric or automatic derivation and is able to use it correctly. And there remains the fact that for numerically difficult densities like, e.g., for the generalized hyperbolic (GH) distribution (see Sect. 5.1) the computational burden of the set-up step is reduced by removing the necessity of the second derivative. It is also likely that for such numerically difficult densities the new version of `Tinflex` is numerically more robust. In addition this approach allows for a more user-friendly implementation of the sampling algorithm. It works also without problem for generating very fast from densities truncated to an arbitrary interval. This is not a simple task as exhibited in [3].

The paper is organized as follows: In Sect. 2 we shortly summarize the sampling method from [6]. In Sect. 3 we present the proposed improvement. Section 4 compiles the entire algorithm, and in Sect. 5 we demonstrate how to apply the new algorithm to generate from the Generalized Inverse Gaussian distribution, from the Generalized Hyperbolic distribution and from the Watson distribution.

2 Transformed Density Rejection with Inflection Points

In this section we summarize the method of [6] and restate the main result in Theorem 1 that is more suitable for our purpose. Moreover, a review of the given proofs show that the conditions for density f can be relaxed as following.

(C1) Density f and thus transformed density $\tilde{f}$ are continuous and piece-wise twice continuously differentiable.

(C2) There is only a finite number of points where $\tilde{f}''$ does not exist. Around each of these points either $\tilde{f}'$ is monotone or $\tilde{f}''$ changes sign. This excludes transformed densities like $\tilde{f}(x) = e^{-|x|}$ while $\tilde{f}(x) = \sqrt[3]{x}$ does work.

(C3) We are given a partition of the domain with finitely many breaking points $-\infty \le b_0 < b_1 < \cdots < b_n < b_{n+1} \le \infty$ where the following holds:

(C3a) In each bounded interval $[b_i, b_{i+1}]$ of the partition the closures of the sets $\{x\colon \tilde{f}''(x) \le 0\}$ and $\{x\colon \tilde{f}''(x) \ge 0\}$ are connected or empty.

(C3b) In each unbounded interval $(-\infty, b_1]$ or $[b_{n-1}, \infty)$, $\tilde{f}$ must be concave and strictly monotone.

Observe that Condition (C3a) holds when there is at most one inflection point in $[b_i, b_{i+1}]$ as stated in the original paper. However, (C3a) also allows transformed densities which are linear on subdomains. Another consequence is that there exists a point $y^* \in (b_l, b_r)$ that separates subdomains $[b_l, y^*]$ and $[y^*, b_r]$ where $\tilde{f}$ is convex and concave, resp., whenever both sets are non-empty. Condition (C3b) is required as otherwise we cannot create a hat function with bounded integral.

Now let $[b_l, b_r]$ be an interval in the domain of density f. We denote the tangent of the transformed density $\tilde{f}$ in the boundary points by $\tilde{t}_l(x) = \tilde{f}(b_l) + \tilde{f}'(b_l)(x - b_l)$ and $\tilde{t}_r(x) = \tilde{f}(b_r) + \tilde{f}'(b_r)(x - b_r)$, resp. Its secant is denoted by $\tilde{r}(x)$ with slope

$$R = \frac{\tilde{f}(b_r) - \tilde{f}(b_l)}{b_r - b_l}\,. \tag{1}$$

In general we use a tilde ˜ to denote functions in transformed scale.

The algorithm is based on the following proposition that immediately follows from [6, Theorems 1 and 2].

Theorem 1 *Let $[b_l, b_r]$ be a bounded closed interval where $\tilde{f}$ satisfies Condition (C3a). Then it belongs to one of the eight types that are listed in Table 2. There are no other types.*

The inequalities in Table 2 allow to determine the type of the interval and to create hat function and squeeze. For types (IVa) and (IVb) both tangents can be used. In [6] the one where $\tilde{f}$ is larger in the respective construction points b_l and b_r is proposed. Figure 1 illustrates three of the possible types.

Table 2 Types of intervals $[b_l, b_r]$

Type	$\tilde{f}'$ and R	$\tilde{f}''$	Squeeze and hat
Ia	$\tilde{f}'(b_l),\ \tilde{f}'(b_r) \geq R$	$\tilde{f}''(b_l) \leq 0 \leq \tilde{f}''(b_r)$	$\tilde{t}_r(x) \leq \tilde{f}(x) \leq \tilde{t}_l(x)$
Ib	$\tilde{f}'(b_l),\ \tilde{f}'(b_r) \leq R$	$\tilde{f}''(b_l) \geq 0 \geq \tilde{f}''(b_r)$	$\tilde{t}_l(x) \leq \tilde{f}(x) \leq \tilde{t}_r(x)$
IIa	$\tilde{f}'(b_l) \geq R \geq \tilde{f}'(b_r)$	$\tilde{f}''(b_l) \leq 0 \leq \tilde{f}''(b_r)$	$\tilde{r}(x) \leq \tilde{f}(x) \leq \tilde{t}_l(x)$
IIb	$\tilde{f}'(b_l) \geq R \geq \tilde{f}'(b_r)$	$\tilde{f}''(b_l) \geq 0 \geq \tilde{f}''(b_r)$	$\tilde{r}(x) \leq \tilde{f}(x) \leq \tilde{t}_r(x)$
IVa	$\tilde{f}'(b_l) \geq R \geq \tilde{f}'(b_r)$	$\tilde{f}''(b_l),\ \tilde{f}''(b_r) \leq 0$	$\tilde{r}(x) \leq \tilde{f}(x) \leq \tilde{t}_l(x), \tilde{t}_r(x)$
IIIa	$\tilde{f}'(b_l) \leq R \leq \tilde{f}'(b_r)$	$\tilde{f}''(b_l) \leq 0 \leq \tilde{f}''(b_r)$	$\tilde{t}_r(x) \leq \tilde{f}(x) \leq \tilde{r}(x)$
IIIb	$\tilde{f}'(b_l) \leq R \leq \tilde{f}'(b_r)$	$\tilde{f}''(b_l) \geq 0 \geq \tilde{f}''(b_r)$	$\tilde{t}_l(x) \leq \tilde{f}(x) \leq \tilde{r}(x)$
IVb	$\tilde{f}'(b_l) \leq R \leq \tilde{f}'(b_r)$	$\tilde{f}''(b_l),\ \tilde{f}''(b_r) \geq 0$	$\tilde{t}_l(x), \tilde{t}_r(x) \leq \tilde{f}(x) \leq \tilde{r}(x)$

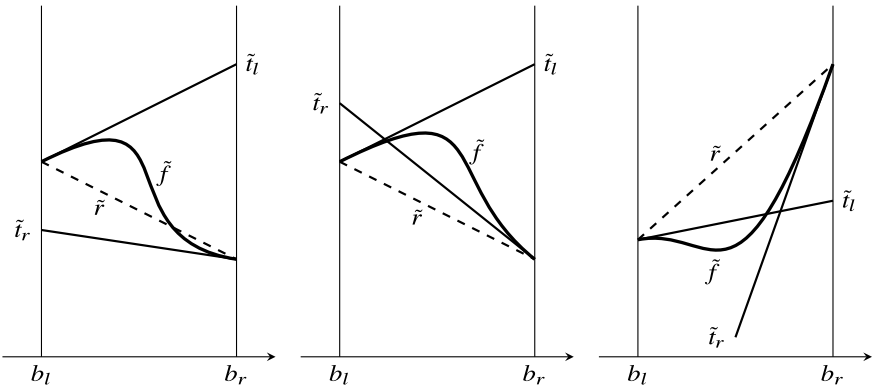


Fig. 1 Intervals of types (Ia), (IIa), and (IIIa) from Table 2

Remark 1 Notice:

- Transformed density $\tilde{f}$ is concave near b_l in types (Ia), (IIa), (IIIa) and (IVa) and convex otherwise. It is concave near b_r in types (Ib), (IIb), (IIIb), and (IVa) and convex otherwise.
- The types in Table 2 are determined by R and the values of $\tilde{f}'$ and $\tilde{f}''$ at boundary points b_l and b_r.
- Tangents $\tilde{t}_l$ and $\tilde{t}_r$ do not intersect in $[b_l, b_r]$ if and only if the interval is of type (Ia) or (Ib).
- Tangents $\tilde{t}_l$ and $\tilde{t}_r$ and secant $\tilde{r}$ form a triangle in $[b_l, b_r]$ with the intersection point of the tangents above the secant if and only if the interval is of type (IIa), (IIIb), or (IVa); and below if and only if the interval is of type (IIb), (IIIa), or (IVb).
- In intervals of type (IVa) and (IVb), $\tilde{f}''$ cannot change sign and thus $\tilde{f}$ is concave and convex, resp., in $[b_l, b_r]$.

It is then straightforward to construct a sampling routine on interval $[b_l, b_r]$. Let

$$\tilde{h} = \alpha + \beta(x - x_0)\,,$$

be the linear hat function for $\tilde{f}$ where x_0 is either b_l or b_r. When the tangent serves as the hat function to $\tilde{f}$, $\alpha = \tilde{f}(x_0)$ and $\beta = \tilde{f}'(x_0)$, and when the secant serves as the hat function to $\tilde{f}$, $\alpha = \tilde{f}(x_0)$ and $\beta = R$. The area below the hat then becomes

$$\begin{aligned} A_h &= \int_{b_l}^{b_r} h(x)\,dx = \int_{b_l}^{b_r} T_c^{-1}(\alpha + \beta(x - x_0))dx \\ &= \frac{1}{\beta}\left[F_T(\alpha + \beta(b_r - x_0)) - F_T(\alpha + \beta(b_l - x_0))\right], \end{aligned} \tag{2}$$

where F_T denotes the anti-derivative of T_c^{-1}. The (non-normalized) CDF, $H(x)$, of the density proportional to h then is given by

$$\begin{aligned} H(x) &= \int_a^x T_c^{-1}(\alpha + \beta(t - x_0))dt \\ &= \frac{1}{\beta}\left[F_T(\alpha + \beta(x - x_0)) - F_T(\alpha + \beta(a - x_0))\right]. \end{aligned}$$

Notice that $H(b_r) = A_h$. Thus the inverse $H^{-1}(u)$ for $u \in [0, A_h]$ is then given by

$$H^{-1}(u) = x_0 + \frac{1}{\beta}\left[F_T^{-1}(\beta u + F_T(\alpha + \beta(a - x_0))) - \alpha\right]. \tag{3}$$

Remark 2 When x_0 is very close to a maximum or another point where $\tilde{f}'(x_0) \approx 0$, then formula (3) becomes sensitive to numerical errors. Then one should replace the exact formula by approximations which are accurate up to the resolution of the floating point numbers, see [6] for details.

Remark 3 When $c \neq 0$ there might be a problem when using tangents to construct hat or squeeze functions. If $\tilde{t}$ has a root in the corresponding interval, then $\tilde{t}$ cannot be transformed back into a valid hat or squeeze function. It is then necessary to further subdivide the corresponding interval as described below.

The performance of an acceptance-rejection method can be measured by means of the rejection constant defined as the ratio of the area below the hat function and the area below the density. It gives the expected number of iterations of the acceptance-rejection loop for getting one (accepted) random variate. An advantage of transformed density rejection as described here is that the rejection constant is bounded by the ratio

$$\rho = \frac{\text{area below hat}}{\text{area below squeeze}}\,. \tag{4}$$

It even allows to estimate the required number of (usually expensive) evaluations of $\tilde{f}$ which is approximately given by $\rho - 1$, see [13]. If ρ is close to one than the marginal generation time hardly depends on the given density.

Another advantage of this method is that ρ can be made as close to 1 as requested by the user. Indeed for the twice continuously differentiable densities with bounded domain we find

$$\rho = 1 + O(1/N^2) \tag{5}$$

when we have N intervals of equal length, see [6].

There exist some methods to find non-overlapping intervals of the domain which result in arbitrary small values of $\rho - 1$. One starts with any subdivision that satisfies Condition (C3). Then this subdivision can be refined by means of adaptive rejection sampling (ARS) as proposed by [11] where rejected points are used to split the corresponding interval into two parts until the requested value of ρ is reached. Alternatively we can iteratively subdivide intervals where the area between hat and squeeze is above some threshold value during the setup, see [15]. We propose to use the"arc-mean" of the boundaries of interval (b_{i-1}, b_i) for splitting intervals:

$$p_{\text{arc}} = \tan\left(\tfrac{1}{2}(\arctan(b_{i-1}) + \arctan(b_i))\right) \tag{6}$$

where $\arctan(\pm\infty)$ is set to $\pm\pi/2$, see also [13, Sect. 4.4.6].

It is then straightforward to compile an algorithm based on the above principles. In Algorithm 4 in Sect. 3 below we present the entire algorithm that makes use of the proposed improvements.

3 Determine Signs of Second Derivatives

We can see from Table 2 that we need the *sign* of $\tilde{f}''$ at the two boundary points only in order to distinguish between types (IIa), (IIb) and (IVa) as well as between types (IIIa), (IIIb) and (IVb). There is no necessity to compute its exact value. So we propose a method that characterizes these types reliably when only the first derivative is available. Again the conditions of Sect. 2 must be satisfied. We discuss two cases:

Case 1: We subdivide an interval of known type and determine the types of the two subintervals. This is useful when the signs of $\tilde{f}''$ are given for the boundaries of the initial intervals and we have to split an interval in order to improve the hat and squeeze. This scenario seems plausible as the user already needs a rough estimate for the inflection points.

Case 2: No such information is given and we have to determine the type of the interval without evaluating the second derivative.

For this purpose we first summarize some basic properties of concave and convex functions. In particular we will make use of properties (a) and (b).

Lemma 1 *Let $\tilde{f}$ be a piece-wise $\mathcal{C}^2$-function on domain $[b_1, b_2]$ such that Condition (C2) holds.*

(a) Point $p^* \in (b_l, b_r)$ is an inflection point of $\tilde{f}$ if and only if p^* is an extremal point of its derivative $\tilde{f}'$.
(b) Derivative $\tilde{f}'$ is monotonically decreasing (increasing) if and only if $\tilde{f}$ is concave (convex).
(c) Function $\tilde{f}$ is concave (convex) if and only if $\tilde{f}''(x) \le 0$ ($\tilde{f}''(x) \ge 0$) for all $x \in [b_1, b_2]$.
(d) Let $\tilde{t}(x)$ be a tangent of $\tilde{f}$.
If $\tilde{f}$ is concave, then $\tilde{f}(x) \le \tilde{t}(x)$ for all $x \in [b_1, b_2]$.
If $\tilde{f}$ is convex, then $\tilde{f}(x) \ge \tilde{t}(x)$ for all $x \in [b_1, b_2]$.
(e) Let $\tilde{t}(x) = \tilde{f}(x_0) + \tilde{f}'(x_0)(x - x_0)$ be the tangent in x_0.
If $\tilde{f}(y) \le \tilde{t}(y)$ for some $y > x_0$, then $\tilde{f}'(x_0) \ge \left(\tilde{f}(y) - \tilde{f}(x_0)\right) / (y - x_0)$.
If $\tilde{f}(y) \le \tilde{t}(y)$ for some $y < x_0$, then $\tilde{f}'(x_0) \le \left(\tilde{f}(y) - \tilde{f}(x_0)\right) / (y - x_0)$.

By our assumptions the sign of the second derivative can be determined by triples of points as stated in the next proposition.

Lemma 2 *Let $\tilde{f}$ be a piecewise $\mathcal{C}^2$-function on domain $[b_1, b_2]$ such that Conditions (C2) and (C3a) hold. Let $b_1 \le p_1 < p_2 < p_3 \le b_2$.*

(a) If $\tilde{f}'(p_2) \le \min\{\tilde{f}'(p_1), \tilde{f}'(p_3)\}$, then $\tilde{f}''(p_1) \le 0 \le \tilde{f}''(p_3)$.
If $\tilde{f}'(p_2) \ge \max\{\tilde{f}'(p_1), \tilde{f}'(p_3)\}$, then $\tilde{f}''(p_1) \ge 0 \ge \tilde{f}''(p_3)$.
(b) If $\tilde{f}'(p_1) \le \tilde{f}'(p_2) \le \tilde{f}'(p_3)$, then $\tilde{f}''(p_2) \ge 0$.
If $\tilde{f}'(p_1) \ge \tilde{f}'(p_2) \ge \tilde{f}'(p_3)$, then $\tilde{f}''(p_2) \le 0$.

Proof By our assumptions there is at most one point or interval where $\tilde{f}'$ is extremal in the open interval (b_1, b_2). If $\tilde{f}'(p_2) \le \min\{\tilde{f}'(p_1), \tilde{f}'(p_3)\}$, then there is exactly one minimum (interval) of $\tilde{f}'$ in (p_1, p_3) and hence $\tilde{f}'$ is decreasing near p_1 and increasing near p_3. Thus (2) follows.

If $\tilde{f}'(p_1) \le \tilde{f}'(p_2) \le \tilde{f}'(p_3)$, then $\tilde{f}'$ is either monotonically increasing, or has a minimum in subinterval (p_1, p_2), or has a maximum in (p_2, p_3). In all cases $\tilde{f}'$ is increasing around p_2 and hence $\tilde{f}''(x_2) \ge 0$ as claimed in (2). □

These elementary tools now enable us to determine $\tilde{f}''$ at the boundary points of the intervals. In Sect. 3.1 we look at the case where no additional information is available. In Sect. 3.2 we split intervals of given types and determine the types of the corresponding subintervals.

3.1 Initial Intervals

Theorem 2 *Let $\tilde{f}$ be a piecewise $\mathcal{C}^2$-function on domain $[b_l, b_r]$ such that Conditions (C2) and (C3a) hold. Let $b_l < p < b_r$ be some point. Then one of the cases in Table 3 holds. No other cases are possible. The properties of the combined types in cases (3.3.3) and (4.3.3) are listed in Table 4.*

Proof Let $p^* \in [b_l, b_r]$ denote the possible inflection point of $\tilde{f}$ (or one of the points that separate the subdomains where $\tilde{f}$ is concave and convex, resp.).

Obviously one of cases (1), (2), (3), or (4) must hold. According to Table 2 cases (1) and (2) determine types (Ia) and (Ib), resp.

Case (3): We have $\tilde{f}'(b_l) \geq R \geq \tilde{f}'(b_r)$. Then by Table 2 interval $[b_l, b_r]$ is of type (IIa), (IIb), or (IVa). In order to distinguish between these cases we look at $\tilde{f}'(p)$. If $\tilde{f}'(p) \leq \tilde{f}'(b_r) = \min\{\tilde{f}'(b_l), \tilde{f}'(b_r)\}$ (case 3.1), then $\tilde{f}''(b_l) \leq 0 \leq \tilde{f}''(b_r)$ by

Table 3 Determination of interval types by means of transformed density $\tilde{f}$ and its derivative $\tilde{f}'$. Point $p \in (b_l, b_r)$ must be an interior point. Symbol (IIIa | IVb) + (IIIb | IVb) means that we have to split the interval at point p into the two subintervals $[b_l, p]$ and $[p, b_r]$

Case	$\tilde{f}'$ and R	$\tilde{f}(p)$	Type
(1)	$\tilde{f}'(b_l), \tilde{f}'(b_r) \geq R$		(Ia)
(2)	$\tilde{f}'(b_l), \tilde{f}'(b_r) \leq R$		(Ib)
(3)	$\tilde{f}'(b_l) \geq R \geq \tilde{f}'(b_r)$		—
(3.1)	$\tilde{f}'(p) \leq \tilde{f}'(b_r)$		(IIa)
(3.2)	$\tilde{f}'(p) \geq \tilde{f}'(b_l)$		(IIb)
(3.3)	$\tilde{f}'(b_l) \geq \tilde{f}'(p) \geq \tilde{f}'(b_r)$		—
(3.3.1)		$\tilde{f}(p) > \tilde{t}_l(p)$	(IIb)
(3.3.2)		$\tilde{f}(p) > \tilde{t}_r(p)$	(IIa)
(3.3.3)		$\tilde{f}(p) \leq \tilde{t}_l(p), \tilde{t}_r(p)$	(IIb | IVa) + (IIa | IVa)
(4)	$\tilde{f}'(b_l) \leq R \leq \tilde{f}'(b_r)$		—
(4.1)	$\tilde{f}'(p) \leq \tilde{f}'(b_l)$		(IIIa)
(4.2)	$\tilde{f}'(p) \geq \tilde{f}'(b_r)$		(IIIb)
(4.3)	$\tilde{f}'(b_l) \leq \tilde{f}'(p) \leq \tilde{f}'(b_r)$		—
(4.3.1)		$\tilde{f}(p) < \tilde{t}_l(p)$	(IIIa)
(4.3.2)		$\tilde{f}(p) < \tilde{t}_r(p)$	(IIIb)
(4.3.3)		$\tilde{f}(p) \geq \tilde{t}_l(p), \tilde{t}_r(p)$	(IIIa | IVb) + (IIIb | IVb)

Table 4 Combined types of intervals $[b_l, b_r]$

Type	$\tilde{f}'$ and R	$\tilde{f}''$	Squeeze and hat
IIa | IVa	$\tilde{f}'(b_l) \geq R \geq \tilde{f}'(b_r)$	$\tilde{f}''(b_l) \leq 0$	$\tilde{r}(x) \leq \tilde{f}(x) \leq \tilde{t}_l(x)$
IIb | IVa	$\tilde{f}'(b_l) \geq R \geq \tilde{f}'(b_r)$	$\tilde{f}''(b_r) \leq 0$	$\tilde{r}(x) \leq \tilde{f}(x) \leq \tilde{t}_r(x)$
IIIa | IVb	$\tilde{f}'(b_l) \leq R \leq \tilde{f}'(b_r)$	$\tilde{f}''(b_r) \geq 0$	$\tilde{t}_r(x) \leq \tilde{f}(x) \leq \tilde{r}(x)$
IIIb | IVb	$\tilde{f}'(b_l) \leq R \leq \tilde{f}'(b_r)$	$\tilde{f}''(b_l) \geq 0$	$\tilde{t}_l(x) \leq \tilde{f}(x) \leq \tilde{r}(x)$

Lemma 2(2) and thus $[b_l, b_r]$ is of type (IIa). Analogously, if $\tilde{f}'(p) \geq \tilde{f}'(b_l) = \max\{\tilde{f}'(b_l), \tilde{f}'(b_r)\}$ (case 3.2), then $[b_l, b_r]$ is of type (IIb).

Otherwise, we have $\tilde{f}'(b_l) \geq \tilde{f}'(p) \geq \tilde{f}'(b_r)$ (case 3.3) and by Lemma 2(2) $\tilde{f}''(p) \leq 0$. We then compare $\tilde{f}'(p)$ to the values of the two tangents $\tilde{t}_l(p)$ and $\tilde{t}_r(p)$. Again we have three subcases. If $\tilde{f}(p) > \tilde{t}_l(p)$ (case 3.3.1), then $\tilde{f}$ cannot be concave in $[b_l, p]$ and thus inflection point p^* is contained in subinterval $[b_l, p]$. In particular we find $\tilde{f}''(b_l) \geq 0 \geq \tilde{f}''(p)$ and $0 \geq \tilde{f}''(b_r)$ which implies that $[b_l, b_r]$ is of type (IIb). Similarly, if $\tilde{f}(p) > \tilde{t}_r(p)$ (case 3.3.2), then we have $\tilde{f}''(p) \leq 0 \leq \tilde{f}''(b_r)$ and thus $[b_l, b_r]$ is of type (IIa).

Otherwise we have $\tilde{f}'(b_l) \geq R \geq \tilde{f}'(b_r)$, $\tilde{f}'(b_l) \geq \tilde{f}'(p) \geq \tilde{f}'(b_r)$, $\tilde{f}''(p) \leq 0$, and $\tilde{f}(p) \leq \min\{\tilde{t}_l(p), \tilde{t}_r(p)\}$ (case 3.3.3). In this case we split the given interval into subintervals $[b_l, p]$ and $[p, b_r]$. There is at most one (inflection) point y^* in each subinterval. Hence $\tilde{f}$ is either concave in $[b_l, p]$ (type IVa) or there is an inflection point with $\tilde{f}''(b_l) \geq 0 \geq \tilde{f}''(p)$. As $\tilde{f}(p) \leq \tilde{t}_l(p)$ we then have $R_l = \frac{\tilde{f}(p) - \tilde{f}(b_l)}{p - b_l} \leq \tilde{f}'(b_l)$. We also have $R_l \geq \tilde{f}'(p)$ since otherwise we had type (Ia) and $\tilde{f}''(b_l) < 0$. Consequently, if there is an inflection point we have type (IIb). Thus we have type (IVa) or (IIb). We denote this combined type by (IIb | IVa). Similarly we find that $[p, b_r]$ is of type (IIa | IVa).

Case 4 with $\tilde{f}'(b_l) \leq R \leq \tilde{f}'(b_r)$ follows analogously to case 3. □

Remark 4 Note that as we do not use $\tilde{f}''$ we can not decide where $\tilde{f}$ is concave or convex in the cases 3.3.3 and 4.3.3 of Table 3. This can only occur for intervals of the starting partition. We therefore always have to split intervals of the starting partition for which $\tilde{f}$ is convex or concave. Nevertheless, it is still possible to use tangents and secants for creating hat and squeeze in such intervals.

3.2 Splitting Intervals

Once we have information about the sign of $\tilde{f}''$ at b_l and b_r we can derive its sign at possible cutting points when we want to refine the partitioning of the domain. The

Table 5 Signs of $\tilde{f}''$ at cutting points c and $c_\delta = c + \delta$ for each type of interval

Type				Splitting point	
Ia, IIa, IIIa	$\tilde{f}''(b_l) \le 0 \le \tilde{f}''(b_r)$,	$\tilde{f}'(c) \le \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(c_\delta) \ge 0$	
		$\tilde{f}'(c) \ge \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(c) \le 0$	
Ib, IIb, IIIb	$\tilde{f}''(b_l) \ge 0 \ge \tilde{f}''(b_r)$,	$\tilde{f}'(c) \le \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(c) \ge 0$	
		$\tilde{f}'(c) \ge \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(c_\delta) \le 0$	
IVa	$\tilde{f}''(b_l), \tilde{f}''(b_r) \le 0$		$\Rightarrow$	$\tilde{f}''(c) \le 0$	
IVb	$\tilde{f}''(b_l), \tilde{f}''(b_r) \ge 0$		$\Rightarrow$	$\tilde{f}''(c) \ge 0$	
IIa | IVa	$\tilde{f}''(b_l) \le 0$,	$\tilde{f}'(c) \le \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(c_\delta) \ge 0$,	$\tilde{f}''(b_r) \ge 0$
		$\tilde{f}'(c) \ge \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(c) \le 0$	
IIIb | IVb	$\tilde{f}''(b_l) \ge 0$,	$\tilde{f}'(c) \le \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(c) \ge 0$	
		$\tilde{f}'(c) \ge \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(c_\delta) \le 0$,	$\tilde{f}''(b_r) \le 0$
IIb | IVa	$\tilde{f}''(b_r) \le 0$,	$\tilde{f}'(c) \le \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(c) \ge 0$,	$\tilde{f}''(b_l) \ge 0$
		$\tilde{f}'(c) \ge \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(c_\delta) \le 0$	
IIIa | IVb	$\tilde{f}''(b_r) \ge 0$,	$\tilde{f}'(c) \le \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(c_\delta) \ge 0$	
		$\tilde{f}'(c) \ge \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(c) \le 0$,	$\tilde{f}''(b_l) \le 0$

following proposition allows to calculate the sign of $\tilde{f}''$ at a cutting point c. The type of the two subintervals then can be determined by means of the derivatives of $\tilde{f}'$ at the new boundary points and the slope R of the secant according to Tables 2 and 4. However, it might be necessary to shift a given cutting point to some point $c_\delta = c + \delta$ for some small $\delta > 0$.

Theorem 3 *Let $\tilde{f}$ be a piecewise $\mathcal{C}^2$-function on domain $[b_l, b_r]$ such that Conditions (C2) and (C3a) hold. Assume that the signs of $\tilde{f}''(b_l)$ or $\tilde{f}''(b_r)$ are known. Then the signs of $\tilde{f}''$ at points $b_l < c < c_\delta < b_r$ are determined as given by Table 5.*

Proof For the proof we restate these implications in Table 6 in a more condensed "raw" form. Table 5 then follows immediately from the characterizations of the corresponding types. Condition "$\tilde{f}''(b_l) \le 0$" in Table 6 means (in abuse of language) that $\tilde{f}$ is either concave on $[b_l, b_r]$ or there exists a $y \in (b_l, b_r)$ such that $\tilde{f}$ is concave on $[b_l, y^*]$ and convex on $[y^*, b_r]$. Analogously for the other three cases. Because of Condition (C3a) one of these cases applies.

Case (1)—corresponds to types (Ia), (IIa), (IIIa), (IVa), and (IIa | IVa): As $\tilde{f}'$ is concave near b_l there is at most one minimum (or interval of minimums) and no

Table 6 Signs of $\tilde{f}''$ at cutting points c and $c_\delta = c + \delta$

(1)	$\tilde{f}''(b_l) \le 0,$	$\tilde{f}'(c) \le \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(b_l) \le 0,$	$\tilde{f}''(c_\delta) \ge 0,$	$\tilde{f}''(b_r) \ge 0$
		$\tilde{f}'(c) \ge \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(b_l) \le 0,$	$\tilde{f}''(c) \le 0$	
(2)	$\tilde{f}''(b_l) \ge 0,$	$\tilde{f}'(c) \le \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(b_l) \ge 0,$	$\tilde{f}''(c) \ge 0$	
		$\tilde{f}'(c) \ge \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(b_l) \ge 0,$	$\tilde{f}''(c_\delta) \le 0,$	$\tilde{f}''(b_r) \le 0$
(3)	$\tilde{f}''(b_r) \le 0,$	$\tilde{f}'(c) \le \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(b_l) \ge 0,$	$\tilde{f}''(c) \ge 0,$	$\tilde{f}''(b_r) \le 0$
		$\tilde{f}'(c) \ge \tilde{f}'(c_\delta)$	$\Rightarrow$		$\tilde{f}''(c_\delta) \le 0,$	$\tilde{f}''(b_r) \le 0$
(4)	$\tilde{f}''(b_r) \ge 0,$	$\tilde{f}'(c) \le \tilde{f}'(c_\delta)$	$\Rightarrow$		$\tilde{f}''(c_\delta) \ge 0,$	$\tilde{f}''(b_r) \ge 0$
		$\tilde{f}'(c) \ge \tilde{f}'(c_\delta)$	$\Rightarrow$	$\tilde{f}''(b_l) \le 0,$	$\tilde{f}''(c) \le 0,$	$\tilde{f}''(b_r) \ge 0$

maximum of $\tilde{f}'$ in interior (b_l, b_r). Now if $\tilde{f}'(c) \le \tilde{f}'(c_\delta)$, then $\tilde{f}'(c_\delta) \le \tilde{f}'(b_r)$ and hence $\tilde{f}''(c_\delta) \ge 0$ by Lemma 2(2). Moreover, $\tilde{f}''(b_r) \ge 0$ as $\tilde{f}''$ cannot change sign in $[c_\delta, b_r]$. Otherwise we have $\tilde{f}'(c) \ge \tilde{f}'(c_\delta)$. Then $\tilde{f}'(b_l) \ge \tilde{f}'(c)$ and hence $\tilde{f}''(c) \le 0$ by Lemma 2(2).

Cases (2)–(4) follow completely analogously. □

Remark 5 The exact value of shifting δ is not crucial as we are only interested in the sign of $\tilde{f}''(c)$ in opposition to methods for numerical derivation. Although we want to replace our choice of cutting point c by one which is quite close, δ need not be very small so that we can avoid possible round-off errors. So, e.g., the choice $\delta = |b_r - b_l|/1000$ is fine.

4 The Algorithm

Now we can compile an algorithm that is based on the results of this paper. Algorithm 1 presents Algorithm `Tinflex-2` when $c = 0$, i.e., when $T_c(x) = \log(x)$. It is obvious that this algorithm can easily be generalized for arbitrary transformations T_c from Table 1. It is quite straight-forward to compute $T_c(f(x))$ and its derivative from $f(x)$ or $\log(f(x))$ and their corresponding derivatives.

For $c < 0$, however, one must check whether a tangent results in a valid (bounded) hat function. Otherwise, the corresponding interval has to be split. This can be implemented by setting the area in such intervals to $A_{h,i} = \infty$. Although one should also check that $\tilde{f}$ is concave and strictly monotone in the possibly unbounded intervals $(-\infty, b_1]$ and $[b_{n-1}, \infty)$ of the given starting partition, in practice it is only necessary that there is at most one inflection point of $\tilde{f}$ within each of them. It is then quite easy to detect an inflection point in one of these unbounded intervals since then the

Algorithm 1: Algorithm `Tinflex-2`

Input: Log-density $\tilde{f}$ with domain (b_l, b_r) and its derivative $\tilde{f}'$
together with partition $b_l = b_0 < b_1 < \ldots < b_{n-1} < b_n = b_r$
that satisfy Conditions (C1)–(C3);
maximal accepted value for $\rho_{\max}$.
Optional: Types of intervals $[b_i, b_{i+1}]$.

Output: Random variate X with density f.

`// Setup: Initial intervals`
1 **for** $i = 0, \ldots, n$ **do**
2 Compute $\tilde{f}(b_i)$ and $\tilde{f}'(b_i)$.
3 **forall the** *intervals* $[b_i, b_{i+1}]$ **do**
4 Determine type of interval using Table 3 (if not provided).
5 Compute intercepts α and slopes β of hat $\tilde{h}_i$ and squeeze $\tilde{s}_i$ using Tables 2 and 4.
6 Compute area $A_{h,i}$ below hat and area $A_{s,i}$ below squeeze using formula (2).

`// Setup: Derandomized adaptive rejection sampling`
7 **repeat**
8 $A_h \leftarrow \sum A_{h,i}$ and $A_s \leftarrow \sum A_{s,i}$.
9 $\bar{A} \leftarrow (A_h - A_s)/(\#\text{ intervals})$.
10 **forall the** *intervals with* $(A_{h,i} - A_{s,i}) > \bar{A}$ **do**
11 Split interval using "arc-mean" (6).
12 Determine type of interval using Tables 5, 2 and 4.
13 Compute hat, squeeze and areas for the two new intervals.
14 **until** $A_h/A_s \leq \rho_{\max}$.

`// Generation`
15 **loop**
16 Generate J with probability vector proportional to $(A_{h,1}, A_{h,2}, \ldots)$.
17 Generate X with density prop. to h_J using formula (3).
18 Generate $U \sim U(0, 1)$.
19 **if** $U\,h(X) \leq s(X)$ **then** `// evaluate squeeze`
20 **return** X.
21 **else if** $U\,h(X) \leq \exp(\tilde{f}(X))$ **then** `// evaluate density`
22 **return** X.
23 **else**
24 Repeat.

construction of a hat function fails. In such cases, we set $A_{h,i} = \infty$, and in the next cycle of derandomized adaptive rejection sampling (Steps 7–14), the interval will be split.

An advantage of the proposed algorithm is that the intervals can be treated independently from each other, i.e., we virtually have distinct and possibly different densities within each of the mutually exclusive intervals which make up the domain. The proposed algorithm thus allows (mostly) arbitrary values of c which may differ on different intervals of the starting partition. Another advantage of the proposed

algorithm is that it works for any multiple of a density of f. There is thus no need to compute a normalization constant.

Also note that Step 16 can be executed in constant time (i.e., independent of the number of intervals) by means of the alias method or the guide table method, see, e.g., [13, Sect. 3].

It is also possible to replace Steps 7–14 (*derandomized* adaptive rejection sampling) by adaptive rejection sampling. However, in opposition to the method proposed by [11] a rejected point is usually not a good choice for splitting an interval. So we suggest to use the point from (6) instead in the interval.

We have coded a proof-of-concept implementation of Algorithm `Tinflex-2` and added a ready-to-use version to our **R** package `Tinflex` [14].

Algorithm `Tinflex-2` is well suited for the fixed parameter setting where we wish to simulate a large number of IID draws from the same density. In particular the marginal generation times hardly depend on the distribution when ρ is (very) close to 1.

A drawback of Algorithm `Tinflex-2` is its time consuming setup step. Thus it may not be the first choice in the dynamic parameter setting where parameters of the distribution are changed from one call of the generator to the next. But it is possible to choose a rather large value of ρ and thus reduce the overhead of the setup part at the expense of increased marginal generation time. In addition one may start with few initial intervals and continue with adaptive rejection sampling. In fact, Gilks and Wild [11] have invented ARS for this dynamic parameter case required for Gibbs sampling, but we have not implemented ARS in our **R** package.

5 Applications

The main advantage of Algorithm `Tinflex-2` is that it can generate from arbitrary distributions with continuous densities. It is only necessary to provide a function that evaluates the log-density and its derivative together with a starting partition into intervals that all include at most one inflection point of the transformed density. We here illustrate the use of `Tinflex-2` applying it to some practical relevant distribution families for which random variate generation is difficult. A first main result is that `Tinflex-2` worked without problem for all distributions we tried. In addition we observed that for $c = -0.5$ the sampling is significantly faster than for $c = 0$.

Botts et al. [6] discuss how Tinflex can be used to generate from the Generalized Inverse Gaussian (GIG) distribution. They explain especially how the mode and the minimum of the log-concavity can be used to form the starting partition for the set-up of Tinflex. In this contribution we show three more examples. In Sect. 5.1 we apply the method to the Generalized Hyperbolic distribution which is important in financial simulations but has a rather cumbersome density. In Sect. 5.2 we discuss the problem of truncated distributions. And finally we present in Sect. 5.3 an example from spatial statistics.

5.1 Generalized Hyperbolic Distribution

The Generalized Hyperbolic (GH) distribution introduced by Barndorff-Nielsen [1] is a very flexible distribution family with semi-heavy tails. It is the mean-variance mixture of a normal and a GIG distribution and popular especially for return modeling in finance. Its density is proportional to

$$f(x) = e^{\beta(x-\mu)} \frac{\mathrm{K}_{\lambda-1/2}\left(\alpha\sqrt{\delta^2+(x-\mu)^2}\right)}{\left(\sqrt{\delta^2+(x-\mu)^2}/\alpha\right)^{1/2-\lambda}},$$

where $\mathrm{K}_\nu(.)$ denotes the modified Bessel function of the third kind. Maybe that complicated density is the reason why there seems to be no rejection algorithm for the GH-distribution available in the literature. Instead most authors suggest to generate GH variates using its mean-variance-mixture representation. It is clear that this generation can not be very fast as a variate of the GIG distribution, a normal variate and a square root are necessary for that approach.

We find it remarkable that our new algorithm `Tinflex-2` can be applied to the GH distribution family to realize a direct generation algorithm. First it is necessary to implement the log-density of the GH distribution and its derivative. This requires the derivative of the modified Bessel function of the third kind, which is equal to $\mathrm{K}_\nu(x)' = -\mathrm{K}_{\nu-1}(x) - (\nu/x)\,\mathrm{K}_\nu(x)$. It is possible to show that for $c = -0.5$ the transformed density $\tilde{f}(x)$ is concave everywhere or can have on one or both sides of the mode a single interval where $\tilde{f}(x)$ is convex. To find the starting partition without using the second derivative it is easiest to use a numeric search algorithm that finds the minimum of the slope of secants of $\tilde{f}'$. If that minimum is positive the transformed density $\tilde{f}$ is concave and we can use the mode and an arbitrary point right and left of the mode as starting partition. If the minimum of the slopes of the secants is negative on one or both sides of the mode the points where these minima are attained can be used together with the mode to form the starting partition.

To test the random variates generated with `Tinflex-2` extensively we used $c = -0.5$ and $\rho_{\max} = 1.001$. For 3850 different parameter values we made the chi-square test with sample-size one million. The histogram of those 3850 P-values (see Fig. 2) confirms that the P-values follow the uniform distribution and thus implies that the generated samples follow the correct distribution.

For $c = -0.5$, $\rho = 1.001$ and a sample of size ten million the generation time, including the set-up, is on our standard laptop around 0.36 s, compared to 0.5 s for the standard **R**-command `rnorm()`. And this speed of `Tinflex-2` for generating from the GH distribution is not influenced by the GH-parameters selected. This is worth to note as it is not the case for GH generation methods based on the GIG generator of Dagpunar [7]. There the running times become extremely slow when $|\lambda| < 0.5$ and parameters α and δ are (very) close to 0. But also for all other parameter settings we have tested the execution time for generating a sample of size ten million is above 1.6 s for the R-libraries `ghyp`, `fBasics` and `GeneralizedHyperbolic`.

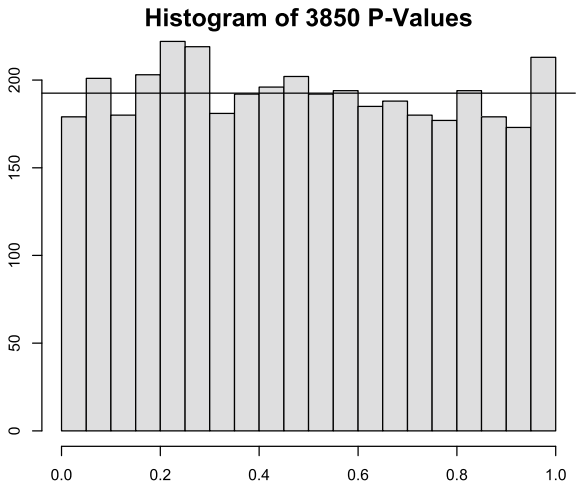


Fig. 2 P-values of Chi-Square tests for 3850 different parameter settings, each with $n = 10^6$

We can conclude that using `Tinflex-2` we can generate large samples from the GH distribution at least 4 times faster than using the standard algorithm suggested in the literature.

5.2 Truncated Distributions

One general advantage of transformed density rejection is that it can be used easily for truncated distributions. To generate truncated random variables from the GH distribution we can use directly the approach explained in Sect. 5.1 above. It is only necessary to find a starting partition of the truncation interval such that each subinterval holds at most one inflection point of the transformed density. Clearly that can be done using the starting partition P_R that was obtained for the whole real line by first removing from P_R all points that are not in the truncation interval and then adding the lower and upper border of the truncation interval to the remaining points. That means of course that if no point of P_R is in the truncation interval the starting partition for the truncated distribution is only the truncation interval itself. Note that this allows to generate without any problems from the truncated GH distribution with arbitrary truncation intervals. We observed in our experiments that the generation of samples of size ten million from the truncated GH-distribution takes, like for the non-truncated case, approximately 0.36 s. Our stable implementation of the log-density of GH and its derivative allows us to generate from truncation intervals in the far tails with extremely small probabilities. For example for $\lambda = .3, \alpha = .2, \beta = 0.02, \delta = .01$ and $\mu = 0$ we generated large samples of the GH-distribution for several different truncation intervals including the interval (1000, 1005). We are not aware of any

paper or software that describes the efficient generation of random variates from the GH-distribution truncated to arbitrary intervals.

We also generated large samples from the truncated normal and from the truncated gamma distribution with shape parameter larger than one and different truncation intervals. As expected there occurred no problems and generating ten million variates takes less than 0.35 s; again clearly less than generating from the normal distribution (0.5 s) or from the gamma distribution (between 0.66 and 1.1 s) using the usual **R**-commands. That the efficient generation of truncated standard distributions is practically relevant and not trivial to achieve, can be seen from two recent papers of Botev and L'Ecuyer [3, 4] implemented in **R** package `TruncatedNormal` [2].

5.3 Watson Distributions

The Watson distribution is used in the modeling of axially symmetric data in spatial statistics. A random unit length vector $\mathbf{X}$ in $\mathbb{R}^d$ has a Watson distribution with concentration parameter $\kappa \in \mathbb{R}$ and mean direction parameter $\boldsymbol{\mu} \in \mathbb{R}^d$ (with $\|\boldsymbol{\mu}\|_2 = 1$) if its density is proportional to $f(\mathbf{x}) \propto \exp\left(\kappa \boldsymbol{\mu}'\mathbf{x}\right)$. We refer to [16] and the literature cited therein for more details. For sampling from this multivariate distribution we can use the identity that for $\boldsymbol{\mu} = (0, \ldots, 0, 1)$, $\mathbf{X} = \left(\sqrt{1 - W^2}\mathbf{Y}, W\right)$, where $\mathbf{Y}$ is uniformly distributed on the hypersphere orthogonal to $\boldsymbol{\mu}$ and W has log-density

$$g(w) = \kappa w^2 + \frac{d-3}{2} \log\left(1 - w^2\right)$$

on domain $[0, 1]$. It is straightforward to verify that $g(w)$ has at most one inflection point and thus `Tinflex-2` can be applied with its domain as starting partition. Sablica et al. [16] have already shown that the predecessor from [6] can be used for this purpose. However, the new version works as well with about the same marginal running time (which is quite similar to the two examples above) but has a simpler user interface.

6 Conclusions

The algorithm presented in this paper is a user-friendly adaptive acceptance-rejection algorithm. It is user-friendly in the sense that hat and squeeze functions of f are constructed automatically without the user having to know the exact location of the inflection points of the transformed density. The only input required from the user is the transformation T_c (in practice in most cases $c = 0$ or $c = -0.5$), the log-density and its derivative and a partition of the domain of f such that the transformed density does not have more than one inflection point in any of the sub-intervals. The new

algorithm improves the method of [6] in the sense that there is no necessity to compute and implement the second derivative of the (log-) density. Our experiments show that the new algorithm is well suited to generate random variates from the Generalized Inverse Gaussian distribution, from the Generalized Hyperbolic distribution and from the Watson distribution. Also random variates from truncated distributions can be generated without problems.

References

1. Barndorff-Nielsen, O.E.: Exponentially decreasing distributions for the logarithm of particle size. Proc. R. Soc. Lon. A **353**, 401–419 (1977)
2. Botev, Z., Belzile, L.: TruncatedNormal: Truncated Multivariate Normal and Student Distributions (2021). https://CRAN.R-project.org/package=TruncatedNormal. R package version 2.2.2
3. Botev, Z., L'Ecuyer, P.: Simulation from the Normal distribution truncated to an interval in the tail. In: 10th EAI International Conference on Performance Evaluation Methodologies and Tools. ACM (2017). https://doi.org/10.4108/eai.25-10-2016.2266879
4. Botev, Z.I., L'Ecuyer, P.: Efficient probability estimation and simulation of the truncated multivariate Student-t distribution. In: 2015 Winter Simulation Conference (WSC), pp. 380–391 (2015). https://doi.org/10.1109/WSC.2015.7408180
5. Botts, C.: A modified adaptive accept-reject algorithm for univariate densities with bounded support. J. Stat. Comput. Simul. **81**(3), 1039–1053 (2011)
6. Botts, C., Hörmann, W., Leydold, J.: Transformed density rejection with inflection points. Stat. Comput. **23**(2), 251–260 (2012). https://doi.org/10.1007/s11222-011-9306-4
7. Dagpunar, J.S.: An easily implemented generalised inverse Gaussian generator. Commun. Stat. - Simul. Comput. **18**(2), 703–710 (1989). https://doi.org/10.1080/03610918908812785
8. Devroye, L.: A simple algorithm for generating random variates with a log-concave density. Computing **33**(3–4), 247–257 (1984)
9. Devroye, L.: Non-Uniform Random Variate Generation. Springer, New-York (1986)
10. Evans, M., Swartz, T.: Random variable generation using concavity properties of transformed densities. J. Comput. Graph. Stat. **7**(4), 514–528 (1998)
11. Gilks, W.R., Wild, P.: Adaptive rejection sampling for Gibbs sampling. Appl. Stat. **41**(2), 337–348 (1992)
12. Hörmann, W.: A rejection technique for sampling from T-concave distributions. ACM Trans. Math. Softw. **21**(2), 182–193 (1995)
13. Hörmann, W., Leydold, J., Derflinger, G.: Automatic Nonuniform Random Variate Generation. Springer, Berlin (2004)
14. Leydold, J., Botts, C., Hörmann, W.: Tinflex: A Universal Non-Uniform Random Number Generator (2022). https://CRAN.R-project.org/package=Tinflex. R package version 2.1
15. Leydold, J., Janka, E., Hörmann, W.: Variants of transformed density rejection and correlation induction. In: Fang, K.T., Hickernell, F.J., Niederreiter, H. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2000, pp. 345–356. Springer, Heidelberg (2002)
16. Sablica, L., Hornik, K., Leydold, J.: Random sampling from the Watson distribution. Research Report Series/Department of Statistics and Mathematics 134, WU Vienna University of Economics and Business, Vienna (2022). https://epub.wu.ac.at/8582/

Fast Automatic Bayesian Cubature Using Sobol' Sampling

Rathinavel Jagadeeswaran and Fred J. Hickernell

Abstract Automatic cubatures approximate integrals to user-specified error tolerances. For high dimensional problems, it is difficult to adaptively change the sampling pattern to focus on peaks because peaks can hide more easily in high dimensional space. But, one can automatically determine the sample size, n, given a reasonable, fixed sampling pattern. This approach is pursued in Jagadeeswaran and Hickernell, Stat. Comput., 29:1214–1229, 2019, where a Bayesian perspective is used to construct a credible interval for the integral, and the computation is terminated when the half-width of the interval is no greater than the required error tolerance. Our earlier work employs integration lattice sampling, and the computations are expedited by the fast Fourier transform because the covariance kernels for the Gaussian process prior on the integrand are chosen to be shift-invariant. In this chapter, we extend our fast automatic Bayesian cubature to digital net sampling via *digitally* shift-invariant covariance kernels and fast Walsh transforms. Our algorithm is implemented in the MATLAB Guaranteed Automatic Integration Library (GAIL) and the QMCPy Python library.

Keywords Adaptive multivariate cubature · Probabilistic numerics · Digital nets · Stopping criteria · GAIL · QMCPy

1 Introduction

Cubature, or numerical multivariate integration, is the problem of inferring a numerical value for a definite integral,

R. Jagadeeswaran (✉)
Department of Applied Mathematics, Illinois Institute of Technology, 10 W. 32nd St., Room 220, Chicago, IL 60616, USA
e-mail: jrathin1@iit.edu

F. J. Hickernell
Center for Interdisciplinary Scientific Computation and Department of Applied Mathematics, Illinois Institute of Technology, 10 W. 32nd St., Room 220, Chicago, IL 60616, USA
e-mail: hickernell@iit.edu

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_15

$$\mu := \mu(f) := \int_{[0,1]^d} f(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}, \tag{1}$$

when no closed-form analytic expression exists or is easily available. Typically, values of f are accessible through a black-box function routine. Our goal is to construct a cubature, $\widehat{\mu}_n = \widehat{\mu}_n(f)$, depending only on integrand values at the nodes $\{\boldsymbol{x}_i\}_{i=1}^n$, and determine the n that satisfies the error criterion

$$|\mu - \widehat{\mu}_n| \le \varepsilon \tag{2}$$

with high probability. This article extends the fast Bayesian cubature ideas presented in [12] to digital sequences [8].

Cubature is a key component of many problems in scientific computing, finance [10], statistical modeling, imaging [14], uncertainty quantification, and machine learning [11]. The original form of the integral may require a suitable variable transformation to become (1). This process is addressed in [1, 6, 15, 22, 23].

Following the Bayesian numerics approach of [3, 7, 19, 20] and others, we assume that our integrand, f, is an instance of a Gaussian process, $\mathcal{GP}(m, s^2 C_{\boldsymbol{\theta}})$, and construct a probabilistic error bound for μ via a Bayesian credible interval. Here, the random function f has constant mean, m, and covariance kernel $s^2 C_{\boldsymbol{\theta}}$, where s is a positive scale factor, and $C_{\boldsymbol{\theta}} : [0,1]^d \times [0,1]^d \to \mathbb{R}$ is a symmetric, positive-definite kernel parameterized by $\boldsymbol{\theta}$. The parameter $\boldsymbol{\theta}$ may affect the shape or smoothness of $C_{\boldsymbol{\theta}}$. The integrand is sampled until the credible interval becomes small enough to satisfy (2) with high probability.

Our approach to *fast* Bayesian cubature [12] relies on two key points:

(i) Choosing covariance kernels, $C_{\boldsymbol{\theta}} : [0,1]^d \times [0,1]^d \to \mathbb{R}$, for which the symmetric, positive definite Gram matrices,

$$\mathsf{C}_{\boldsymbol{\theta}} = \big(C_{\boldsymbol{\theta}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\big)_{i,j=1}^n, = (\boldsymbol{C}_{\boldsymbol{\theta},1}, \ldots, \boldsymbol{C}_{\boldsymbol{\theta},n}), \tag{3}$$

have an eigenvalue-eigenvector decomposition of the form[1] $\mathsf{C}_{\boldsymbol{\theta}} = \mathsf{V}\Lambda_{\boldsymbol{\theta}}\mathsf{V}^H/n$, where

$$\mathsf{V} \text{ may be identified analytically,} \tag{4a}$$

$$\text{The first row and column of } \mathsf{V} \text{ are } \mathbf{1}, \tag{4b}$$

$$\text{Computing } \mathsf{V}^H \boldsymbol{b} \text{ requires only } \mathcal{O}(n \log n) \text{ operations } \forall \boldsymbol{b}, \text{ and} \tag{4c}$$

$$\int_{[0,1]^d} C_{\boldsymbol{\theta}}(\boldsymbol{t}, \boldsymbol{x}) \, \mathrm{d}\boldsymbol{t} = 1 \qquad \forall \boldsymbol{x} \in [0,1]^{\mathrm{d}}, \tag{4d}$$

and

[1] The presence of $1/n$ in the eigenvalue-eigenvector decomposition arises from the assumption that the first column of V is $\mathbf{1}$. It could be removed by assuming that the first column of V is $\mathbf{1}/\sqrt{n}$. The superscript H denotes the complex conjugate transpose [2].

(ii) The hyperparameters of the Gaussian process—m, s, and $\boldsymbol{\theta}$—are tuned to increase the likelihood that f is a typical integrand and not an outlier.

We call the transformation $\boldsymbol{b} \mapsto \mathsf{V}^H \boldsymbol{b}$ a *fast Bayesian transform*. Our earlier work [12] focuses on lattice nodes and shift-invariant kernels. In that context the computation of $\mathsf{V}^H \boldsymbol{b}$ is a one-dimensional fast Fourier transform. This chapter focuses on digital sequences, such as Sobol' sequences [24], and covariance kernels that are digitally shift-invariant. In this case the computation of $\mathsf{V}^H \boldsymbol{b}$ is a fast Walsh-Hadamard transform.

The next section summarizes the key formulae from [12]. Section 3 extends fast Bayesian cubature to digital nets and digital-shift invariant kernels defined in terms of Walsh functions. Section 4 presents numerical experiments that illustrate these ideas.

2 Bayesian Cubature

As noted above, we assume the integrand, f, is an instance of a real-valued stochastic Gaussian process, i.e., $f \sim \mathcal{GP}(m, s^2 C_{\boldsymbol{\theta}})$. Let $\boldsymbol{y} = (y_i)_{i=1}^n = \big(f(\boldsymbol{x}_i)\big)_{i=1}^n$ denote the vector of sampled integrand values. In [12] we introduce three methods for resolving the hyperparameters: empirical Bayes (EB), full Bayes (full), and generalized cross-validation (GCV). Under assumptions (4a)–(4d) for the covariance kernel and sampling sites, it is shown in [12, Theorem 2] that the credible interval for the integral takes the form

$$\mathbb{P}_f \left[|\mu - \widehat{\mu}_n| \le \mathrm{err}_{\mathrm{CI}} \right] = 99\%,$$

where

$$\widehat{\mu}_n = \frac{1}{n} \sum_{i=1}^n y_i = \widetilde{y}_1, \quad \text{where } y_i = f(\boldsymbol{x}_i), \tag{5}$$

$$\widetilde{\boldsymbol{y}} := \mathsf{V}^H \boldsymbol{y}, \quad \text{where } \widetilde{\boldsymbol{y}} \text{ is the fast transform of } \boldsymbol{y},$$

$$\boldsymbol{\lambda}_{\boldsymbol{\theta}} := \begin{pmatrix} \lambda_{\boldsymbol{\theta},1} \\ \vdots \\ \lambda_{\boldsymbol{\theta},n} \end{pmatrix} = \mathsf{V}^H \boldsymbol{C}_{\boldsymbol{\theta},1},$$

where $\boldsymbol{C}_{\boldsymbol{\theta},1}$ is the first column of the Gram matrix (3),

$$s_{\mathrm{EB}}^2 = \frac{1}{n^2} \sum_{i=2}^n \frac{|\widetilde{y}_i^2|}{\lambda_{\boldsymbol{\theta},i}},$$

$$s_{\mathrm{GCV}}^2 = \frac{1}{n} \sum_{i=2}^n \frac{|\widetilde{y}_i^2|}{\lambda_{\boldsymbol{\theta},i}^2} \left[\sum_{i=1}^n \frac{1}{\lambda_{\boldsymbol{\theta},i}} \right]^{-1},$$

$$\widehat{\sigma}^2_{\text{full}} = \frac{1}{n(n-1)} \sum_{i=2}^{n} \frac{|\widetilde{y}_i^2|}{\lambda_{\boldsymbol{\theta},i}} \left(\frac{\lambda_{\boldsymbol{\theta},1}}{n} - 1 \right),$$

$$\boldsymbol{\theta}_{\text{EB}} = \operatorname*{argmin}_{\boldsymbol{\theta}} \left[\log \left(\sum_{i=2}^{n} \frac{|\widetilde{y}_i^2|}{\lambda_{\boldsymbol{\theta},i}} \right) + \frac{1}{n} \sum_{i=1}^{n} \log(\lambda_{\boldsymbol{\theta},i}) \right], \tag{6a}$$

$$\boldsymbol{\theta}_{\text{GCV}} = \operatorname*{argmin}_{\boldsymbol{\theta}} \left[\log \left(\sum_{i=2}^{n} \frac{|\widetilde{y}_i^2|}{\lambda_{\boldsymbol{\theta},i}^2} \right) - 2 \log \left(\sum_{i=1}^{n} \frac{1}{\lambda_{\boldsymbol{\theta},i}} \right) \right]. \tag{6b}$$

$$\text{err}_{\text{CI},\mathsf{x}} = \text{err}_{\mathsf{x}} = 2.58 s_{\mathsf{x}} \sqrt{1 - \frac{n}{\lambda_{\boldsymbol{\theta}_{\mathsf{x}},1}}}, \qquad \mathsf{x} \in \{\text{EB}, \text{GCV}\}, \tag{7a}$$

$$\text{err}_{\text{CI,full}} = t_{n-1,0.995} \widehat{\sigma}_{\text{full}} > \text{err}_{\text{EB}}. \tag{7b}$$

In the formulas for the credible interval half-widths, $\boldsymbol{\theta}$ is assumed to take on the values $\boldsymbol{\theta}_{\text{EB}}$ or $\boldsymbol{\theta}_{\text{GCV}}$ as appropriate. There is no suitable $\boldsymbol{\theta}_{\text{full}}$.

Our Bayesian cubature algorithm doubles the sample size until the width of the credible interval is small enough. Doubling the sample size allows us to retain the preferred structure of the sample sites. The Bayesian cubature steps are detailed in Algorithm 1.

Algorithm 1: Automatic Bayesian Cubature

Input: a generator for the sequence $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$; a black-box function, f; an absolute error tolerance, $\varepsilon > 0$; the positive initial sample size, n_0; the maximum sample size $n_{\max}$

$n \leftarrow n_0$, $n' \leftarrow 0$, $\text{err}_{\text{CI}} \leftarrow \infty$;
while $\text{err}_{\text{CI}} > \varepsilon$ *and* $n \le n_{\max}$ **do**
  Generate $\{\boldsymbol{x}_i\}_{i=n'+1}^{n}$ and sample $\{f(\boldsymbol{x}_i)\}_{i=n'+1}^{n}$;
  Compute $\boldsymbol{\theta}$ by (6a) or (6b);
  Compute err_{CI} according to (7a) or (7b);
  $n' \leftarrow n$, $n \leftarrow 2n'$;
Sample size to compute $\widehat{\mu}_n$, $n \leftarrow n'$;
Compute $\widehat{\mu}_n$, the approximate integral, according to (5);
return $\widehat{\mu}_n$, *n, and* err_{CI};

We recognize that multiple applications of our credible intervals in one run of the algorithm is not strictly justified. However, if our integrand comes from the middle of the sample space and not the extremes, we expect our automatic Bayesian cubature to approximate the integral within the desired error tolerance with high probability. The examples in Sect. 4 support that expectation.

The credible intervals in our automatic algorithm are homogeneous with respect to the function data. If they are valid for some integrand, f, they are also valid for the integrand af for any constant a.

3 Digital Nets and Walsh Kernels

The previous section does not mention which sequences of data sites and kernels satisfy assumptions (4a, 4b, 4c, 4d). We demonstrate in [12] that rank-1 lattice points and shift-invariant kernels do so. In this section, we give another example, namely digital sequences and digitally shift-invariant kernels based on Walsh functions. For completeness, we define digital sequences, digitally shift-invariant kernels, and the fast Walsh-Hadamard transform.

3.1 Digital Sequences

The first example of a digital sequence was proposed by Sobol' [24], and it is also the most popular digital sequence. This chapter focuses on digital sequences in base 2, which includes Sobol' sequences.

Definition 1 Let $(\,)_2$ denote the base 2 or binary expansion of a number. For any non-negative integer $i = (\ldots i_3 i_2 i_1)_2$, define $\overrightarrow{\imath} = (i_1, i_2, \ldots)^T$ as the $\infty \times 1$ vector $\overrightarrow{\imath}$ of its binary digits. For any point $z = (0.z_1 z_2 \ldots)_2 \in [0, 1)$, define $\overrightarrow{z} = (z_1, z_2, \ldots)^T$ as the $\infty \times 1$ vector of its binary digits. Let $\mathsf{G}_1, \ldots, \mathsf{G}_d$ denote predetermined $\infty \times \infty$ generator matrices whose elements are zeros and ones. A *digital sequence* in base 2 is $\{\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3, \ldots\}$, where each $\boldsymbol{z}_i = (z_{i1}, \ldots, z_{id})^T \in [0, 1)^d$ is defined by

$$\overrightarrow{z}_{i+1,\ell} = \mathsf{G}_\ell \overrightarrow{\imath} \pmod 2, \quad \ell = 1, \ldots, d, \quad i = 0, 1, \ldots .$$

For any non-negative integers m and τ, the set $\{\boldsymbol{z}_{\tau 2^m+1}, \ldots, \boldsymbol{z}_{(\tau+1)2^m+1}\}$ is a *digital net* in base 2.

It is common to index digital sequences starting with 0, whereas our data sites and matrices are indexed starting with 1. To keep our notation consistent, we define $\boldsymbol{z}_{i+1}$ in terms of $\overrightarrow{\imath}$.

Digital sequences have a group structure under digit-wise, element-by-element addition modulo the base, which we denote by $\oplus$ and which also corresponds to an exclusive-or in base 2. Here and in what follows we ignore the cases of measure zero for which the $\oplus$ operation leads to a binary representation ending in an infinite string of ones. The following lemma summarizes some important properties of digital sequences.

Lemma 1 *Let $\{z_i\}_{i=1}^{\infty}$ be a digital sequence in base 2 as defined in Definition 1. Choose any digital shift $\boldsymbol{\Delta} \in [0,1)^d$, and define the sequence of nodes $\{\boldsymbol{x}_i\}_{i=1}^{\infty}$ by digitwise addition, $\boldsymbol{x}_i = \boldsymbol{z}_i \oplus \boldsymbol{\Delta}$. Then for all $i, j \in \mathbb{N}_0$,*

$$\boldsymbol{x}_{i+1} \oplus \boldsymbol{x}_{i+1} = \boldsymbol{0}, \quad \boldsymbol{x}_{i+1} \oplus \boldsymbol{x}_{j+1} = \boldsymbol{x}_{j+1} \oplus \boldsymbol{x}_{i+1} = \boldsymbol{z}_{i+1} \oplus \boldsymbol{z}_{j+1} = \boldsymbol{z}_{(i \oplus j)+1}.$$

Therefore, $\{\boldsymbol{z}_i\}_{i=1}^{\infty}$ is a group. Moreover, the digital net $\{\boldsymbol{z}_i\}_{i=1}^{2^m}$ is a subgroup for $m \in \mathbb{N}_0$.

The proof follows from Definition 1, and can be found in [21].

Digital sequence generators can be chosen by number theory, as are those of Sobol' [24] and Niederreiter and Xing [17] (see also [8, Chap. 8]), or they can be chosen by computer search [8, Chap. 10]. The original generator matrices may be scrambled using linear matrix scrambling [16].

3.2 Covariance Kernels Constructed Via Walsh Functions

The digitally shift invariant kernels required for fast Bayesian cubature using digital nets are constructed via Walsh functions, again specializing to base 2. The one-dimensional Walsh functions in base 2 are defined as

$$\begin{aligned} \mathrm{wal}_k(x) &:= (-1)^{k_0 x_1 + k_1 x_2 + \cdots} = (-1)^{\langle \vec{k}, \vec{x} \rangle}, \qquad k \in \mathbb{N}_0, \ x \in [0,1), \\ \langle \vec{k}, \vec{x} \rangle &:= k_0 x_1 + k_1 x_2 + \cdots . \end{aligned}$$

where again $\vec{k}$ is a vector containing the binary digits of k, and $\vec{x}$ is a vector containing the binary digits of x. Note that by this definition, $\mathrm{wal}_k(x \oplus t) = \mathrm{wal}_k(x)\mathrm{wal}_k(t)$.

These Walsh functions can be used to construct a covariance kernel for univariate integrands as follows

$$\begin{aligned} C_{\boldsymbol{\theta}}(x,t) &= K_{\boldsymbol{\theta}}(x \oplus t), \qquad K_{\boldsymbol{\theta}}(x) := 1 + \eta \omega_r(x), \qquad \boldsymbol{\theta} = (r, \eta), \\ \omega_r(x) &:= \sum_{k=1}^{\infty} \frac{\mathrm{wal}_k(x)}{2^{2r\lfloor \log_2 k \rfloor}}, \qquad r \in \mathbb{N}, \qquad \eta \in (0,1) \\ \omega_r(x_\ell \oplus t_\ell) &= \sum_{k=1}^{\infty} \frac{\mathrm{wal}_k(x)\mathrm{wal}_k(t)}{2^{2r\lfloor \log_2 k \rfloor}}. \end{aligned}$$

The symmetric, positive definite property of $C_{\boldsymbol{\theta}}(x,t)$ follows from its definition. This kernel is digitally shift invariant because

$$C_{\boldsymbol{\theta}}(x \oplus \Delta, t \oplus \Delta) = K_{\boldsymbol{\theta}}(x \oplus \Delta \oplus t \oplus \Delta) = K_{\boldsymbol{\theta}}(x \oplus t) = C_{\boldsymbol{\theta}}(x,t).$$

This follows because $\Delta \oplus \Delta = 0$ for any $\Delta \in [0,1)$.

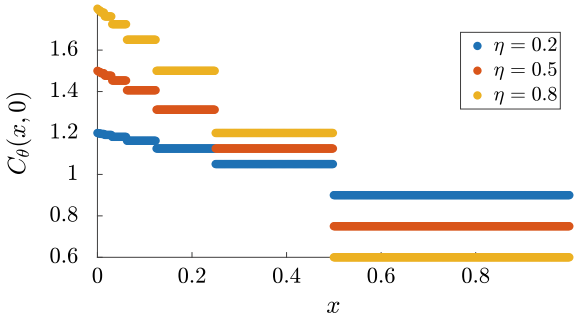


Fig. 1 Walsh kernel of order $r = 1$ in dimension $d = 1$. This figure can be reproduced using `plot_walsh_kernel.m`

The parameter r is a measure of the digital smoothness of the kernel, which does not correspond to ordinary smoothness. An explicit expression is available for ω_r in the case of order $r = 1$ (see [18]):

$$\omega_1(x) = 1 - 6 \times 2^{\lfloor \log_2 x \rfloor - 1}.$$

Figure 1 shows $C_{\boldsymbol{\theta}}(x, t)$ for order $r = 1$ and various values of η in the interval $[0, 1)$. Unlike the shift-invariant kernels used with lattice nodes, Walsh kernels are discontinuous and piecewise constant.

Covariance kernels for multivariate integrands defined on $[0, 1)^d$ are constructed as tensor products:

$$C_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{t}) = K_{\boldsymbol{\theta}}(\boldsymbol{x} \oplus \boldsymbol{t}), \tag{8}$$

$$K_{\boldsymbol{\theta}}(\boldsymbol{x}) = \prod_{\ell=1}^{d} [1 + \eta_\ell \omega_r(x_\ell)], \quad \boldsymbol{\eta} = (\eta_1, \ldots, \eta_d), \quad \boldsymbol{\theta} = (r, \boldsymbol{\eta}). \tag{9}$$

For multidimensional kernels, smaller η_ℓ for $\ell \in (1, \ldots, d)$ implies less variation in the amplitude of the kernel in dimension ℓ. The parameter vector $\boldsymbol{\theta}$ now is of dimension $d + 1$. One might also choose $\eta_1 = \cdots = \eta_d = \eta$, in which case the parameter vector $\boldsymbol{\theta} = (r, \eta)$ has dimension two.

3.3 *Eigenvector-Eigenvalue Decomposition of the Gram Matrix*

For fast Bayesian cubature to succeed, the digital net data sites (Sect. 3.1) and the covariance kernels (Sect. 3.2) must match in a way to satisfy conditions (4a, 4b, 4c, 4d). To do this, we notice that the eigenvectors of the Gram matrix defined in (3) are the columns of the Walsh-Hadamard matrices, defined as follows:

$$\mathsf{H}^{0)} = 1, \quad \mathsf{H}^{(1)} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad \mathsf{H}^{(2)} = \mathsf{H}^{(1)} \otimes \mathsf{H}^{(1)} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}, \ldots$$

$$\mathsf{H}^{(m)} = \mathsf{H}^{(m-1)} \otimes \mathsf{H}^{(1)} = \begin{pmatrix} \mathsf{H}^{(m-1)} & \mathsf{H}^{(m-1)} \\ \mathsf{H}^{(m-1)} & -\mathsf{H}^{(m-1)} \end{pmatrix} = \mathsf{H}^{(1)} \underbrace{\otimes \cdots \otimes}_{m \text{ times}} \mathsf{H}^{(1)}, \tag{10}$$

where $\otimes$ is Kronecker product. Note that the Walsh-Hadamard matrices are symmetric.

Lemma 2 *Let $\{\boldsymbol{x}_i\}_{i=1}^{2^m}$ be nodes from a digitally shifted digital net in base 2, and let the covariance kernel take the form of (8). Then, for $m = 1, 2, \ldots$ the Gram matrix, $\mathsf{C}_{\boldsymbol{\theta}}^{(m)} = \big(C_{\boldsymbol{\theta}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\big)_{i,j=1}^{2^m} = \big(K(\boldsymbol{x}_i \oplus \boldsymbol{x}_j)\big)_{i,j=1}^{2^m}$ is a 2×2 block-Hankel matrix [2, Definition 3.1.3], and all the sub-blocks and their sub-sub-blocks, etc. are also 2×2 block-Hankel. Moreover, $\mathsf{C}_{\boldsymbol{\theta}}^{(m)} \mathsf{H}^{(m)} = \mathsf{H}^{(m)} \Lambda_{\boldsymbol{\theta}}^{(m)}$, where $\Lambda_{\boldsymbol{\theta}}^{(m)}$ is the diagonal matrix of eigenvalues of $\mathsf{C}_{\boldsymbol{\theta}}^{(m)}$.*

Proof. First define the following matrices using the notation from Lemma 1:

$$\begin{aligned} \mathsf{C}_{\boldsymbol{\theta}}^{(m,\tau)} &= \big(K(\boldsymbol{x}_i \oplus \boldsymbol{x}_{j+\tau 2^m})\big)_{i,j=1}^{2^m} \\ &= \big(K(\boldsymbol{z}_{i \oplus j+\tau 2^m+1})\big)_{i,j=0}^{2^m-1}, \qquad m, \tau = 0, 1, \ldots. \end{aligned}$$

This implies that $\mathsf{C}_{\boldsymbol{\theta}}^{(m+1,\tau)}$, has the following block structure:

$$\begin{aligned} &\mathsf{C}_{\boldsymbol{\theta}}^{(m+1,\tau)} \\ &\quad = \begin{pmatrix} \big(K(\boldsymbol{z}_{i \oplus j+\tau 2^{m+1}+1})\big)_{i,j=0}^{2^m-1} & \big(K(\boldsymbol{z}_{i \oplus (j+2^m)+\tau 2^{m+1}+1})\big)_{i,j=0}^{2^m-1} \\ \big(K(\boldsymbol{z}_{(i+2^m) \oplus j+\tau 2^{m+1}+1})\big)_{i,j=0}^{2^m-1} & \big(K(\boldsymbol{z}_{(i+2^m) \oplus (j+2^m)+\tau 2^{m+1}+1})\big)_{i,j=0}^{2^m-1} \end{pmatrix} \\ &\quad = \begin{pmatrix} \mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau)} & \mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau+1)} \\ \mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau+1)} & \mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau)} \end{pmatrix}, \end{aligned} \tag{11}$$

since for $i, j = 0, \ldots, 2^m - 1$, it follows that

$$\begin{aligned} i \oplus (j + 2^m) &= (i \oplus j) + 2^m = (i + 2^m) \oplus j, \\ (i + 2^m) \oplus (j + 2^m) &= (i \oplus j) + (2^m \oplus 2^m) = i \oplus j. \end{aligned}$$

The proof of (11) follows by induction. Note that for the case $m = 1$ and $\tau = 0, 1, \ldots$

$$\mathsf{C}_{\boldsymbol{\theta}}^{(1,\tau)} = \begin{pmatrix} \mathsf{C}_{\boldsymbol{\theta}}^{(0,2\tau)} & \mathsf{C}_{\boldsymbol{\theta}}^{(0,2\tau+1)} \\ \mathsf{C}_{\boldsymbol{\theta}}^{(0,2\tau+1)} & \mathsf{C}_{\boldsymbol{\theta}}^{(0,2\tau)} \end{pmatrix} = \begin{pmatrix} K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+1}) & K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+2}) \\ K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+2}) & K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+1}) \end{pmatrix},$$

which has the desired Hankel property. Note also the eigenvectors of $\mathsf{C}_{\boldsymbol{\theta}}^{(1,\tau)}$ are the columns of $\mathsf{H}^{(1)}$ since

$$\begin{aligned}\mathsf{C}_{\boldsymbol{\theta}}^{(1,\tau)}\mathsf{H}^{(1)} &= \begin{pmatrix} K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+1}) + K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+2}) & K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+1}) - K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+2}) \\ K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+2}) + K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+1}) & K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+2}) - K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+1}) \end{pmatrix} \\ &= \mathsf{H}^{(1)} \underbrace{\begin{pmatrix} K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+1}) + K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+2}) & 0 \\ 0 & K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+1}) - K_{\boldsymbol{\theta}}(\boldsymbol{z}_{2\tau+2}) \end{pmatrix}}_{\Lambda_{\boldsymbol{\theta}}^{(1,\tau)}}.\end{aligned}$$

Now assume that $\mathsf{C}_{\boldsymbol{\theta}}^{(m,\tau)}$ is 2×2 block-Hankel matrix with all its sub-blocks and sub-sub-blocks, etc. also 2×2 block-Hankel for $\tau = 0, 1, \ldots$. Then by (11), the same holds for $\mathsf{C}_{\boldsymbol{\theta}}^{(m+1,\tau)}$ for $\tau = 0, 1, \ldots$.

Moreover, the hypothesized eigenvectors of $\mathsf{C}_{\boldsymbol{\theta}}^{(m+1,\tau)}$ can also be verified by direct calculation under the induction hypothesis:

$$\begin{aligned}&\mathsf{C}_{\boldsymbol{\theta}}^{(m+1,\tau)}\mathsf{H}^{(m+1)} \\ &\quad= \begin{pmatrix} \mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau)} & \mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau+1)} \\ \mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau+1)} & \mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau)} \end{pmatrix} \begin{pmatrix} \mathsf{H}^{(m)} & \mathsf{H}^{(m)} \\ \mathsf{H}^{(m)} & -\mathsf{H}^{(m)} \end{pmatrix} \\ &\quad= \begin{pmatrix} [\mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau)} + \mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau+1)}]\mathsf{H}^{(m)} & [\mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau)} - \mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau+1)}]\mathsf{H}^{(m)} \\ [\mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau+1)} + \mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau)}]\mathsf{H}^{(m)} & [\mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau+1)} - \mathsf{C}_{\boldsymbol{\theta}}^{(m,2\tau)}]\mathsf{H}^{(m)} \end{pmatrix} \\ &\quad= \begin{pmatrix} \mathsf{H}^{(m)} & \mathsf{H}^{(m)} \\ \mathsf{H}^{(m)} & -\mathsf{H}^{(m)} \end{pmatrix} \underbrace{\begin{pmatrix} \Lambda_{\boldsymbol{\theta}}^{(m,2\tau)} + \Lambda_{\boldsymbol{\theta}}^{(m,2\tau+1)} & 0 \\ 0 & \Lambda_{\boldsymbol{\theta}}^{(m,2\tau)} - \Lambda_{\boldsymbol{\theta}}^{(m,2\tau+1)} \end{pmatrix}}_{\Lambda_{\boldsymbol{\theta}}^{(m+1,2\tau)}}.\end{aligned}$$

Noting that $\mathsf{C}_{\boldsymbol{\theta}}^{(m+1)} = \mathsf{C}_{\boldsymbol{\theta}}^{(m+1,0)}$ completes the proof.

The theoretical results of this chapter can be summarized as follows:

Theorem 1 *Any symmetric, positive definite, digitally shift-invariant kernel of the form* (9) *scaled to satisfy* (4d)*, when matched with digital sequence data sites, satisfies assumptions (4a, 4b, 4c, 4d). The* fast Walsh-Hadamard transform *performs the fast Bayesian transform,* $\boldsymbol{b} \mapsto \mathsf{V}^H\boldsymbol{b}$*, in* $\mathcal{O}(n \log n)$ *operations.*

Proof. Lemma 2 establishes that covariance kernels of the form (8) matched with shifted digital net data sites satisfy the fast Bayesian cubature assumptions

- (4a), since V is the Walsh-Hadamard matrix,
- (4b), since the first column and row of the Walsh-Hadamard matrix consist of all ones, and
- (4d), since all Walsh functions but the zeroth integrate to zero.

What remains to be shown is (4c), namely how $\mathsf{V}^H\boldsymbol{b} = \mathsf{H}\boldsymbol{b}$ can be calculated in $\mathcal{O}(n \log n)$ operations for arbitrary $\boldsymbol{b}$.

The computation of $\widetilde{\boldsymbol{b}}^{(m)} = \mathsf{H}^{(m)}\boldsymbol{b}^{(m)}$ is done iteratively in what we show to be $m2^m$ operations, where $\mathsf{H}^{(m)}$ is $2^m \times 2^m$. The proof proceeds by induction. For the case $m = 0$, the Hadamard matrix $\mathsf{H}^{(0)}$ is the scalar 1, and $\widetilde{\boldsymbol{b}}^{(0)} = \mathsf{H}^{(0)}\boldsymbol{b}^{(0)} = \boldsymbol{b}^{(0)}$, which requires no arithmetic operations. Now, assume that $\mathsf{H}^{(m)}\boldsymbol{b}^{(m)}$ requires $m2^m$ operations, and let $\boldsymbol{b}^{(m+1)} = (\boldsymbol{b}_U^{(m)T}, \boldsymbol{b}_L^{(m)T})^T$. Let $\widetilde{\boldsymbol{b}}^{(m+1)} = \mathsf{H}^{(m+1)}\boldsymbol{b}^{(m+1)}$, $\widetilde{\boldsymbol{b}}_U^{(m)} = \mathsf{H}^{(m)}\boldsymbol{b}_U^{(m)}$, and $\widetilde{\boldsymbol{b}}_L^{(m)} = \mathsf{H}^{(m)}\boldsymbol{b}_L^{(m)}$. It follows from the definition of the Walsh-Hadamard matrix that

$$\begin{aligned}\widetilde{\boldsymbol{b}}^{(m+1)} &= \mathsf{H}^{(m+1)}\boldsymbol{b}^{(m+1)} = \begin{pmatrix} \mathsf{H}^{(m)} & \mathsf{H}^{(m)} \\ \mathsf{H}^{(m)} & -\mathsf{H}^{(m)} \end{pmatrix} \begin{pmatrix} \boldsymbol{b}_U^{(m)} \\ \boldsymbol{b}_L^{(m)} \end{pmatrix} \qquad \text{by (10)} \\ &= \begin{pmatrix} \mathsf{H}^{(m)}\boldsymbol{b}_U^{(m)} + \mathsf{H}^{(m)}\boldsymbol{b}_L^{(m)} \\ \mathsf{H}^{(m)}\boldsymbol{b}_U^{(m)} - \mathsf{H}^{(m)}\boldsymbol{b}_L^{(m)} \end{pmatrix} = \begin{pmatrix} \widetilde{\boldsymbol{b}}_U^{(m)} + \widetilde{\boldsymbol{b}}_L^{(m)} \\ \widetilde{\boldsymbol{b}}_U^{(m)} - \widetilde{\boldsymbol{b}}_L^{(m)} \end{pmatrix}.\end{aligned}$$

Thus, to compute $\widetilde{\boldsymbol{b}}^{(m+1)}$ requires two matrix multiplications by $\mathsf{H}^{(m)}$, at a cost of $m2^m$ each, plus an addition and a subtraction of vectors of length 2^m, for a total cost of $2 \times m2^m + 2 \times 2^m = (m+1)2^{m+1}$, which is exactly what is hypothesized. Since $\widetilde{\boldsymbol{b}}^{(m)} = \mathsf{H}^{(m)}\boldsymbol{b}^{(m)}$ requires $m2^m = n\log_2(n)$ operations to compute, assumption (4c) is satisfied. This completes the proof of Theorem 1. □

4 Numerical Experiments

The Bayesian cubature algorithm described here, using the digitally shift invariant kernel in (8) with order $r = 1$ and the Bayesian lattice cubature algorithm described in [12] have been coded as `cubBayesNet_g` and `cubBayesLattice_g`, respectively, in GAIL [4]. We illustrate our algorithm for three common examples, which are also discussed in [12]. The first example evaluates a multivariate Gaussian probability, the second example is Keister's function [13], and the final example is pricing an Asian arithmetic mean option. For each example we are able to find the true integral value, μ, either by analytic computation, or by running a numerical algorithm with a very small error tolerance.

The nodes used in `cubBayesNet_g` are the randomly scrambled and shifted Sobol' points supplied by MATLAB's Sobol' sequence generator. Four hundred different error tolerances, ε, are randomly chosen such that $\log(\varepsilon)$ has a uniform distribution. Although $\log(\varepsilon)$ is uniformly chosen, the times are stratified since the number of samples must be a power of 2. For each integral example, each ε, and each stopping criterion—empirical Bayes, full Bayes, and generalized cross-validation—we ran `cubBayesNet_g`. For each run, the execution time is plotted against $|\mu - \widehat{\mu}_n|/\varepsilon$. We expect $|\mu - \widehat{\mu}_n|/\varepsilon$ to be no greater than one, but hope that it is not too much smaller than one, which would indicate that the stopping criterion is too conserva-

tive. Throughout the experiments we use $\eta_1 = \cdots = \eta_d = \eta$. All the tests were run in MATLAB version 2019b running on Intel i7-7700HQ.

4.1 Multivariate Gaussian Probability

This integral is formulated as in [12] following the variable transformation introduced by Alan Genz [9]. The problem is originally three-dimensional but is transformed to a $d = 2$ dimensional problem. The simulation results are summarized in Figs. 2, 3, and 4. In all cases, `cubBayesNet_g` returns an approximation within the prescribed error tolerance. For $\varepsilon = 10^{-5}$ with the empirical Bayes stopping criterion, `cubBayesNet_g` takes about 3 s as shown in Fig. 2 whereas using a Matérn kernel requires 30 s to obtain the same accuracy as shown in [12]. This highlights the speed-up possible using fast Bayesian cubature.

The algorithm `cubBayesNet_g` uses MATLAB's fast Walsh transform. This is slower than MATLAB's fast Fourier transform, which is implemented in compiled code. This may help explain why `cubBayesLattice_g` is faster than `cubBayesNet_g` for this example. Also, `cubBayesLattice_g` uses higher order kernels whereas higher order digitally shift-invariant kernels are inappropriate for these examples. On average, `cubBayesLattice_g` uses $n \approx 16{,}000$ samples for $\varepsilon = 10^{-5}$, whereas `cubBayesNet_g` uses $n \approx 32{,}000$ samples. Amongst the three stopping criteria for `cubBayesNet_g`, GCV achieves an acceptable approximation faster than others and is also less conservative.

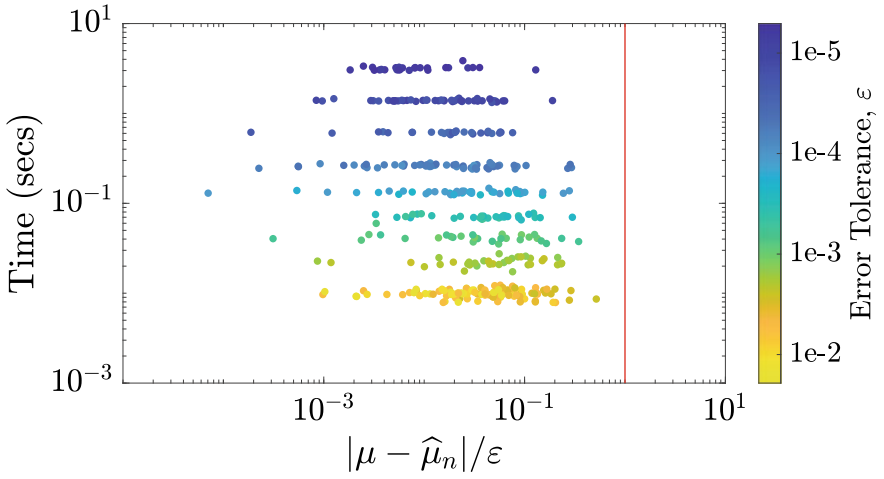


Fig. 2 Multivariate normal probability example with empirical Bayes stopping criterion. Algorithm meets the error threshold for all the ε randomly chosen in $[10^{-5}, 10^{-2}]$

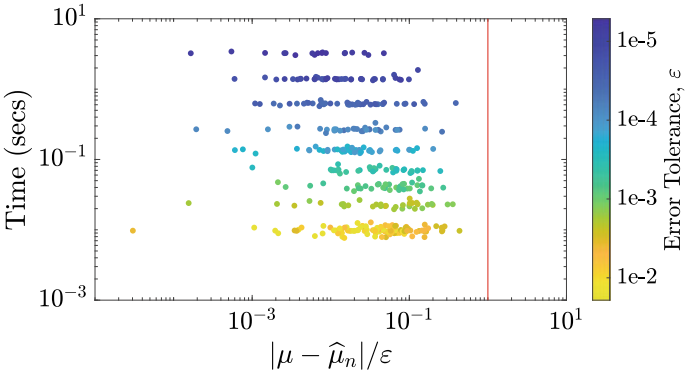


Fig. 3 Multivariate normal probability example with the full-Bayes stopping criterion. Algorithm meets the error threshold for all the ε randomly chosen in $[10^{-5}, 10^{-2}]$

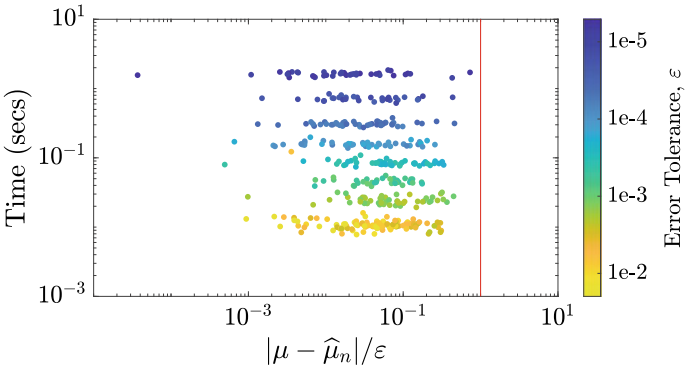


Fig. 4 Multivariate normal probability example with the GCV stopping criterion. Algorithm meets the error threshold for all the ε randomly chosen in $[10^{-5}, 10^{-2}]$

4.2 Keister's Example

This multidimensional integral function comes from [13] and is inspired by a physics application:

$$\mu = \int_{\mathbb{R}^d} \cos(\|\boldsymbol{t}\|) \exp(-\|\boldsymbol{t}\|^2) \, \mathrm{d}\boldsymbol{t} = \int_{[0,1]^d} f_{\text{Keister}}(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x},$$

where

$$f_{\text{Keister}}(\boldsymbol{x}) = \pi^{d/2} \cos\left(\|\boldsymbol{\Phi}^{-1}(\boldsymbol{x})/2\|\right),$$

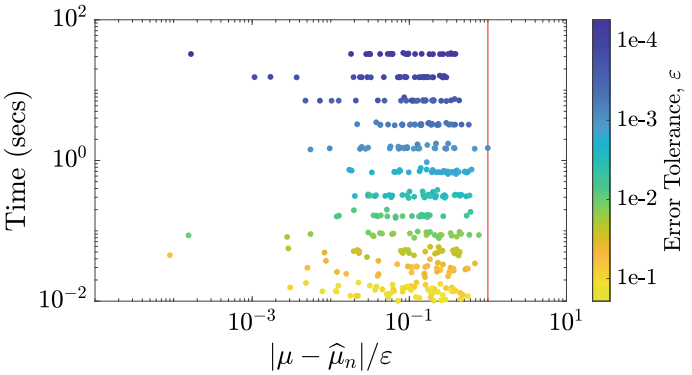


Fig. 5 Keister example using the empirical Bayes stopping criterion. Algorithm meets the error threshold for all the ε randomly chosen in $[10^{-4}, 10^{-1}]$

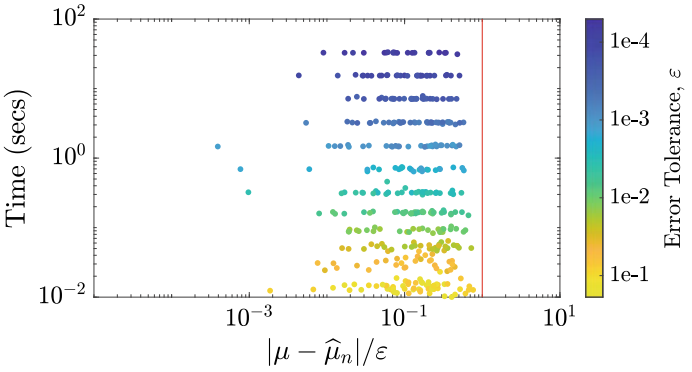


Fig. 6 Keister example using the full-Bayes stopping criterion. Algorithm meets the error threshold for all the ε randomly chosen in $[10^{-4}, 10^{-1}]$

and $\boldsymbol{\Phi}$ is the component-wise standard normal distribution. The true value of μ can be calculated iteratively in terms of quadrature as found in [12, Sect. 5.2].

Figures 5, 6 and 7 summarize the numerical tests for this case for dimension $d = 4$ and order $r = 1$. The `cubBayesLattice_g` algorithm in [12] uses a much smoother kernel than that used here. This explains why `cubBayesNet_g` uses more samples than `cubBayesLattice_g` on average. As observed from the figures, the GCV stopping criterion, in Fig. 7 achieves the results faster than the others but it is less conservative which is also the case with the multivariate Gaussian example.

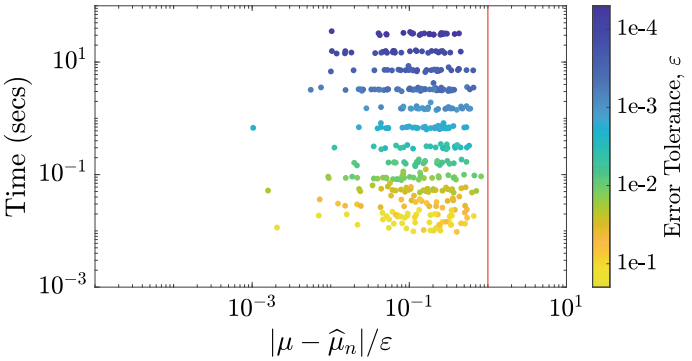


Fig. 7 Keister example using the GCV stopping criterion. Algorithm meets the error threshold for all the ε randomly chosen in $[10^{-4}, 10^{-1}]$

4.3 Asian Option Pricing

The price of financial derivatives can often be modeled by high dimensional integrals. We refer to the formulation of the fair price of the option as in [12], where the underlying asset is described in terms of a discretized geometric Brownian motion.

Figures 8, 9 and 10 summarize the numerical results for the option pricing example using the values for time horizon $T = 1/4$ of a year, $d = 13$ time steps, initial asset price of $S_0 = 100$, interest rate of 0.05 per year, volatility of $\sigma = 0.5$ per root year, and strike price of $K = 200$, the same as used in the experiments with `cubBayesLattice_g` [12]. This integrand has a kink caused by the max function, so lattice rules cannot perform better, even if the integrand were to be periodized. We observe that `cubBayesNet_g` is more efficient in terms of the number

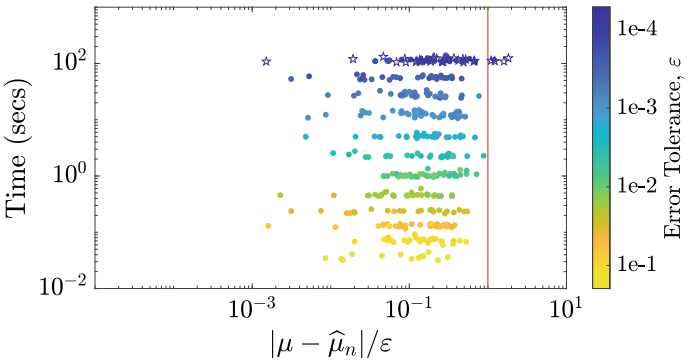


Fig. 8 Option pricing using the empirical Bayes stopping criterion. The 33 hollow stars at the higher time indicate that the half-width of the credible interval did not meet the error threshold ε before reaching maximum n

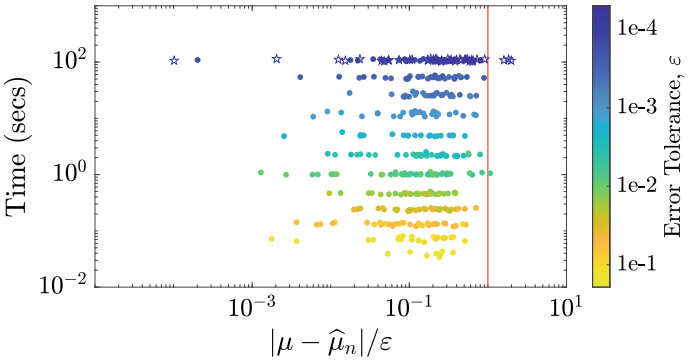


Fig. 9 Option pricing using the full-Bayes stopping criterion. The 41 hollow stars indicate the half-width of the credible interval did not meet the error threshold ε before reaching maximum n

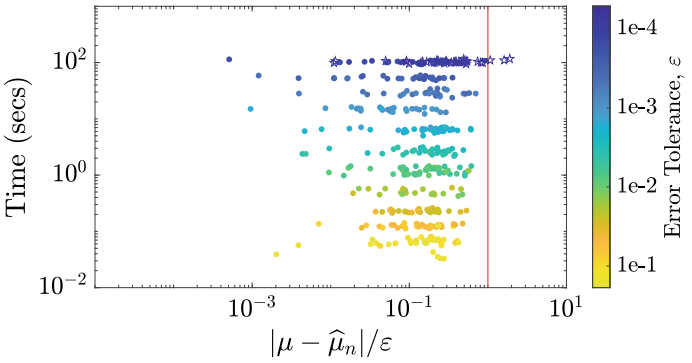


Fig. 10 Option pricing using the GCV stopping criterion. The 35 hollow stars indicate the half-width of the credible interval did not meet the error threshold ε before reaching maximum n

of samples used than `cubBayesLattice_g`. For the error tolerance, $\varepsilon = 10^{-3}$, `cubBayesLattice_g` uses $n \approx 2^{20}$ samples, whereas `cubBayesNet_g` uses $n \approx 2^{17}$ samples.

4.4 Discussion

As shown in Figs. 2, 3, 4, 5, 6, 7, 8, 9 and 10, our algorithm computed the integral within user specified threshold with some exceptions. The exceptions occurred for the option pricing example due to the complexity and higher dimension of the integrand, the small tolerance, and the limit on the sample size. This indicates that fast Bayesian cubature may suffer in such cases and that further study is needed. Also notice that our algorithm `cubBayesNet_g`, finished within 40 s per run for the multivariate

Gaussian and Keister examples for the smallest tolerance as shown in Figs. 2, 3, 4, 5, 6 and 7. Option pricing took little above 100 s per run due to the complexity of integrand as shown in Figs. 8, 9 and 10.

A noticeable aspect from the plots of `cubBayesNet_g` is how close the error bounds are to the true error in many cases. This shows that the `cubBayesNet_g`'s error bounding is not too conservative.

In nearly all of the examples, the ratio $|\mu - \widehat{\mu}_n|/\varepsilon$ is closer to one for the Bayesian net cubature in contrast to the Bayesian lattice cubature. A possible reason is that the periodization transform speeds the convergence in the latter case and our data based error bounds are unable to capture this smaller error.

5 Conclusion and Future Work

We have extended our fast automatic Bayesian cubature to digital net sampling via digitally shift-invariant covariance kernels and fast Walsh transforms. Implementation of our algorithm `cubBayesNet_g`, is available in the Guaranteed Automatic Integration Library (GAIL) [4] and Quasi-Monte-Carlo Software in Python (QMCPy) [5]. We demonstrated `cubBayesNet_g` using three example integrands compared with `cubBayesLattice_g`. One major advantage of this algorithm, unlike the `cubBayesLattice_g` developed in [12], is that the integrand does not have to be periodic. However, unlike `cubBayesNet_g`, `cubBayesLattice_g` is more efficient for smoother, periodic functions when using sufficiently smooth and periodic covariance kernels.

The `cubBayesNet_g` in the current implementation uses only the first order kernel and digital nets. Accuracy and speed of the algorithm could be improved by using higher order digital nets and smoother digitally shift-invariant covariance kernels. This could help with the smoother integrands. This is a promising direction for future work.

For higher dimensions, both Bayesian cubature algorithms sometimes fail to produce an acceptable answer within a reasonable amount of time. This seems to be related to an excessive amount of time required to identify the optimal shape parameter $\boldsymbol{\theta}$. The root of the problem and its resolution is a matter for future investigation.

Acknowledgements We are grateful to Professor Pierre L'Ecuyer for his friendship and many fruitful and enjoyable discussions on Monte Carlo methods. Thanks to the referees for their helpful comments.

References

1. Beckers, M., Haegemans, A.: Transformation of integrands for lattice rules. In: Espelid, T.O., Genz, A.C. (eds.) Numerical Integration: Recent Developments, Software and Applications, pp. 329–340. Kluwer Academic Publishers, Dordrecht (1992)
2. Bernstein, D.S.: Matrix Mathematics: Theory, Facts, and Formulas. Princeton University Press, Princeton and Oxford (2009)
3. Briol, F.X., Oates, C.J., Girolami, M., Osborne, M.A., Sejdinovic, D.: Probabilistic integration: a role in statistical computation? Statist. Sci. **34**, 1–22 (2019)
4. Choi, S.C.T., Ding, Y., Hickernell, F.J., Jiang, L., Jiménez Rugama, L.A., Li, D., Jagadeeswaran, R., Tong, X., Zhang, K., Zhang, Y., Zhou, X.: GAIL: Guaranteed Automatic Integration Library (versions 1.0–2.3.2). MATLAB software (2021). http://gailgithub.github.io/GAIL_Dev/. https://doi.org/10.5281/zenodo.4018189
5. Choi, S.C.T., Hickernell, F.J., Jagadeeswaran, R., McCourt, M., Sorokin, A.: QMCPy: a quasi-Monte Carlo Python library (2020). https://doi.org/10.5281/zenodo.3964489. https://qmcsoftware.github.io/QMCSoftware/
6. Cristea, L.L., Dick, J., Leobacher, G., Pillichshammer, F.: The tent transformation can improve the convergence rate of quasi-Monte Carlo algorithms using digital nets. Numer. Math. **105**, 413–455 (2007)
7. Diaconis, P.: Bayesian numerical analysis. In: Gupta, S.S., Berger, J.O. (eds.) Statistical Decision Theory and Related Topics IV, Papers from the 4th Purdue Symposium, West Lafayette, Indiana 1986, vol. 1, pp. 163–175. Springer, New York (1988)
8. Dick, J., Pillichshammer, F.: Digital Nets and Sequences: Discrepancy Theory and Quasi-Monte Carlo Integration. Cambridge University Press, Cambridge (2010)
9. Genz, A.: Numerical computation of multivariate normal probabilities. J. Comput. Graph. Statist. **1**, 141–150 (1992)
10. Glasserman, P.: Monte Carlo Methods in Financial Engineering, Applications of Mathematics, vol. 53. Springer, New York (2004)
11. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). http://www.deeplearningbook.org
12. Jagadeeswaran, R., Hickernell, F.J.: Fast automatic Bayesian cubature using lattice sampling. Stat. Comput. **29**, 1215–1229 (2019). https://doi.org/10.1007/s11222-019-09895-9
13. Keister, B.D.: Multidimensional quadrature algorithms. Comput. Phys. **10**, 119–122 (1996). https://doi.org/10.1063/1.168565
14. Keller, A.: Quasi-Monte Carlo image synthesis in a nutshell. In: Dick, J., Kuo, F.Y., Peters, G.W., Sloan, I.H. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2012, Springer Proceedings in Mathematics and Statistics, vol. 65, pp. 213–249. Springer, Berlin Heidelberg (2013)
15. Laurie, D.: Periodizing transformations for numerical integration. J. Comput. Appl. Math. **66**, 337–344 (1996)
16. Matoušek, J.: On the L_2-discrepancy for anchored boxes. J. Complex. **14**, 527–556 (1998)
17. Niederreiter, H., Xing, C.: Rational Points on Curves over Finite Fields: Theory and Applications. No. 285 in London Mathematical Society Lecture Note series. Cambridge University Press, Cambridge (2001)
18. Nuyens, D.: The construction of good lattice rules and polynomial lattice rules. In: Uniform Distribution and Quasi-Monte Carlo Methods (2013)
19. O'Hagan, A.: Bayes-Hermite quadrature. J. Statist. Plann. Inference **29**, 245–260 (1991). https://doi.org/10.1016/0378-3758(91)90002-V
20. Rasmussen, C.E., Ghahramani, Z.: Bayesian Monte Carlo. In: Thrun, S., Saul, L.K., Obermayer, K. (eds.) Advances in Neural Information Processing Systems, vol. 15, pp. 489–496. MIT Press (2003)
21. Rathinavel, J.: Fast automatic Bayesian cubature using matching kernels and designs. Ph.D. thesis, Illinois Institute of Technology, Chicago (2019). www.math.iit.edu

22. Sidi, A.: A new variable transformation for numerical integration. In: Brass, H., Hämmerlin, F. (eds.) Numerical Integration IV, no. 112 in International Series of Numerical Mathematics, pp. 359–373. Birkhäuser, Basel (1993)
23. Sidi, A.: Further extension of a class of periodizing variable transformations for numerical integration. J. Comput. Appl. Math. **221**, 132–149 (2008)
24. Sobol', I.M.: The distribution of points in a cube and the approximate evaluation of integrals. U.S.S.R. Comput. Math. and Math. Phys. **7**, 86–112 (1967)

Rendering Along the Hilbert Curve

Alexander Keller, Carsten Wächter, and Nikolaus Binder

Abstract Based on the seminal work on Array-RQMC methods and rank-1 lattice sequences by Pierre L'Ecuyer and collaborators, we introduce efficient deterministic algorithms for image synthesis. Enumerating a low discrepancy sequence along the Hilbert curve superimposed on the raster of pixels of an image, we achieve noise characteristics that are desirable with respect to the human visual system, especially at very low sampling rates. As compared to the state of the art, our simple algorithms neither require randomization, nor costly optimization, nor lookup tables. We analyze correlations of space-filling curves and low discrepancy sequences, and demonstrate the benefits of the new algorithms in a professional, massively parallel light transport simulation and rendering system.

Keywords Quasi-Monte Carlo methods · Hilbert curve · Array-RQMC · Low discrepancy sequences · Rank-1 lattice sequences · Image synthesis

1 Introduction

In photorealistic image synthesis by light transport simulation, the colors of each pixel are an integral of a high-dimensional function. While the functions to integrate are square-integrable and hence of finite energy, they contain discontinuities that cannot be predicted efficiently. In practice, the pixel colors are estimated by Monte Carlo and quasi-Monte Carlo methods sampling light transport paths that connect light sources and cameras and summing up the contributions.

A. Keller (✉) · C. Wächter · N. Binder
NVIDIA, Fasanenstr. 81, 10623 Berlin, Germany
e-mail: akeller@nvidia.com

C. Wächter
e-mail: cwachter@nvidia.com

N. Binder
e-mail: nbinder@nvidia.com

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_16

As a consequence of sampling, images appear noisy when the number of samples is insufficient. This is quite common when images need to be synthesized rapidly for real-time applications and when convergence is slow due to the intricacies of the functions to integrate. Depending on the characteristics of the noise in the image, filtering may efficiently improve image quality.

The success of image compression algorithms and compressive sensing methods clearly indicates that the pixels of an image are not independent integrals. One way to account for the correlation of pixels is to consider image synthesis an integro-approximation problem [12].

In this article we propose a new way of synthesizing images as a sequence of correlated integrals such that noise is less perceivable by the human visual system. We therefore review the state of the art in addressing perceived image error in computer graphics in Sect. 2 and introduce our new deterministic algorithm for image synthesis by enumerating a low discrepancy sequence along the Hilbert Curve in Sect. 3. We then explore extensions for progressive image synthesis in Sect. 4 and discuss the results in Sect. 5 before drawing the conclusions.

For the scope of our article, it is sufficient to understand that the mapping of a vector of the low discrepancy sequence to a light transport path is the same across all discussed methods and that methods only differ in which vector of a low discrepancy sequence is assigned to what pixel. This abstraction allows for reproducing the results. For the experiments, we use the Iray light transport simulation and rendering system [19]. For the details we refer to [4, 7, 15, 19] and extensive background information in [27]. A recent survey of sampling methods in computer graphics is [17], while the latest research focuses on low discrepancy sequences with good low-dimensional projections [25, 26].

2 Visual Error in Image Synthesis

The human visual system is quite capable of recovering information from noisy images, and computer graphics has been taking advantage of that since its early days [5]. Using the same set of samples across the pixels to synthesize an image may result in disturbingly visible aliasing artifacts. Hence, inspired by the arrangement of receptors in a monkey's retina [30], a huge body of work around sampling patterns with blue noise characteristics emerged. Especially at low sampling rates and in low dimensions, these patterns have been attractive since they are close to the ideal of reconstructing precisely as long as the assumptions of the sampling theorem are fulfilled. At the same time, aliases are mapped to noise, which is very amenable to the human visual system.

For long, an important detail had not been considered explicitly: blue noise characteristics of samples do not matter much for a single pixel integral but when observing an ensemble of neighboring pixels. Only recently, it was found that optimizing the parameters of a Cranley-Patterson rotation per pixel applied to one generic set of samples can dramatically improve the perceived image quality although the ℓ^2-error

remains about the same [10]. Subsequent work extends the optimization to scramblings of the Sobol' sequence [13]. The visual improvements have been attributed to blue noise characteristics.

Since the ℓ^2-error between a reference image and different sampling schemes remains about the same, the improvement in perceived image quality must be in the the distribution of the error and how the samples are correlated across the pixels [2].

Based on the family of Array-RQMC methods [21] introduced by Pierre L'Ecuyer, we propose simple deterministic quasi-Monte Carlo algorithms that result in similar visual improvements without optimization. In addition, an explanation for the improvements beyond blue noise characteristics is offered. Our approach benefits from the improved uniformity of low discrepancy sequences observed when simulating Markov chains [21, 29] by ordering their states by proximity in world space. Instead of sorting, we explore orders provided by space filling curves in screen space.

3 Enumerating Pixels Along the Hilbert Curve

Given the resolution of an image to synthesize, a deterministic low discrepancy sequence, and a number of samples to be drawn per pixel, our deterministic algorithm to enumerate the samples per pixel starts by selecting the resolution of the Hilbert curve (see Fig. 1) to match the image resolution. We therefore determine the smallest power of two that is larger or equal to the maximum of the image resolution in horizontal and vertical direction. Enumerating the pixels along the Hilbert curve, for each pixel we draw the selected samples from the low discrepancy sequence in contiguous blocks. Pixels outside the image are simply skipped.

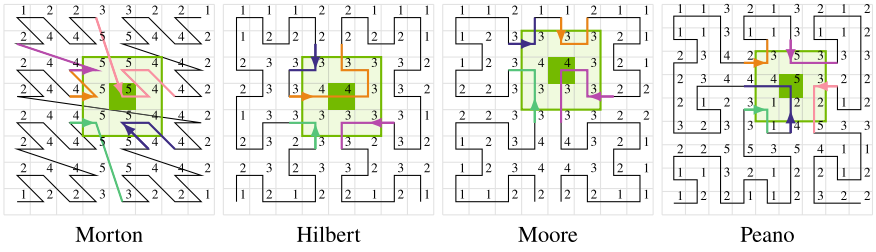


Fig. 1 The Morton, Hilbert, Moore, and Peano space-filling curves on a pixel grid. As the Hilbert, the Moore, and the Peano curve only pass through neighboring pixels, they realize shortest routes of visiting all pixels in the sense of the Traveling Salesman problem. The pixels highlighted in dark green exemplify the number of segments (colored) of each space-filling curve entering the 3×3 neighborhood. This number of segments is depicted for each pixel and its maximum is smallest for the Hilbert and Moore curves. Enumerating a low discrepancy sequence along a space filling curve, a smaller number of segments implies longer contiguous segments of the low discrepancy sequence used in the 3×3 neighborhood, which improves uniformity

As shown in Fig. 5, this simple algorithm performs astonishingly well in comparison to using the first two dimensions of the same low discrepancy sequence to sample the pixels of the image mapped to the unit square [12]. While both approaches expose about the same ℓ^2-error as argued in [10], sampling along the Hilbert curve results in noise that is much more uniformly distributed across the image, especially visible at low sampling rates.

The human visual system always tries to detect scale invariant features and unfortunately finds such in the non-uniformities of noise, too. As such features are not related to the actual image content, they are perceived as disturbing artifacts. If, however, the noise is uniform, it is less likely misinterpreted and consequently noise is not perceived as much. As a result, one may argue that the eye is filtering the noise by integrating over areas of uniform noise in the image. While the blue noise sampling approaches mentioned in the previous section rely on this phenomenon, our new approaches are deterministic and do not require optimization.

Enumerating low discrepancy sequences along space filling curves by spatial proximity [29] suggests that contiguous blocks of samples from a low discrepancy sequence are spatially close and hence improve local uniformity. Similarly, using a variant of the Morton curve (see Fig. 1) combined with scrambling to enumerate pixels [2] results in an error more uniformly distributed across the image. The observable improvements are supported by the fact that the low discrepancy of a point sequence is preserved when enumerated along the Hilbert curve [11]. In addition, we can adopt an argument from [21, Sec.3.2]: Considering an image as a line of pixels as enumerated along the Hilbert curve and assuming the function to be integrated along the pixel to have a gradient bounded by K, the total variation is bounded by K times the length of the Hilbert curve, which in turn bounds the integration error by the Koksma-Hlawka inequality [23]. While the Hilbert, Moore, or Peano curve achieve a shortest route to connect all pixels, the Morton curve fails to do so, which explains parts of its inferiority. On a historical note, both [29] and [2] mention the Hilbert curve but used the Morton curve and hence cannot not take advantage of the above argument. As compared to the Morton and Peano curve, both the Hilbert and Moore curve expose a smaller maximum number of curves segments in the 3×3 neighborhood of a pixel (see Fig. 1), resulting in more consecutive samples of the low discrepancy sequence in the neighborhood, which improves uniformity locally.

In computer graphics, gradients may be bounded in parts of the integration domain. However, such parts usually cannot be identified efficiently. Yet, Fig. 5 clearly shows that in such smoother regions of an image the noise is much more uniformly distributed when using the proposed algorithm. The human visual system takes advantage of these local improvements.

3.1 Correlation in Space-Filling Curves

We show that the correlation patterns visualized in Fig. 2 emanate from enumerating radical inverses [23] along a space-filling curve. A radical inverse

$$\phi_b : \mathbb{N}_0 \to \mathbb{Q} \cap [0, 1)$$
$$i = \sum_{k=0}^{\infty} a_k(i) b^k \mapsto \sum_{k=0}^{\infty} a_k(i) b^{-k-1} \tag{1}$$

maps a non-negative integer to the unit interval by reflecting its digits $a_k(i)$ in base b at the decimal point. The Halton sequence is an example of an infinite-dimensional low discrepancy sequence. Each dimension is a radical inverse, where all the bases

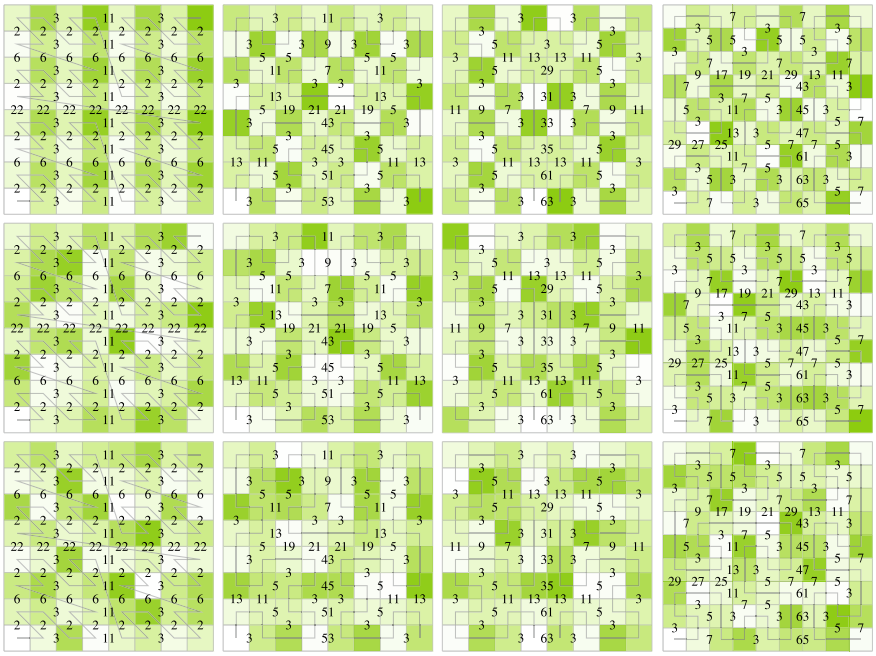


Fig. 2 The numbers on the edges shared by neighboring pixels are the difference of pixel indices enumerated along the depicted space-filling curve. Along the curve, this difference is one and implicitly represented by the imprinted curves. Diagrams in a column belong to the named space-filling curve. From top to bottom, the rows show the differences for the radical inverses ϕ_2, ϕ_3, and ϕ_5, while the shade of each pixel represents the value of the radical inverse of the pixel index. As can be seen, the differences contain symmetries and repetitive patterns, which result in visible correlations in the pixel shades. For curves in base $b = 2$, it is easy to spot a checker-boarding effect. For the Peano curve, symmetries along the diagonal may be observed. Besides this illustration of the principle, structures may be more visible at higher resolutions, see Fig. 3

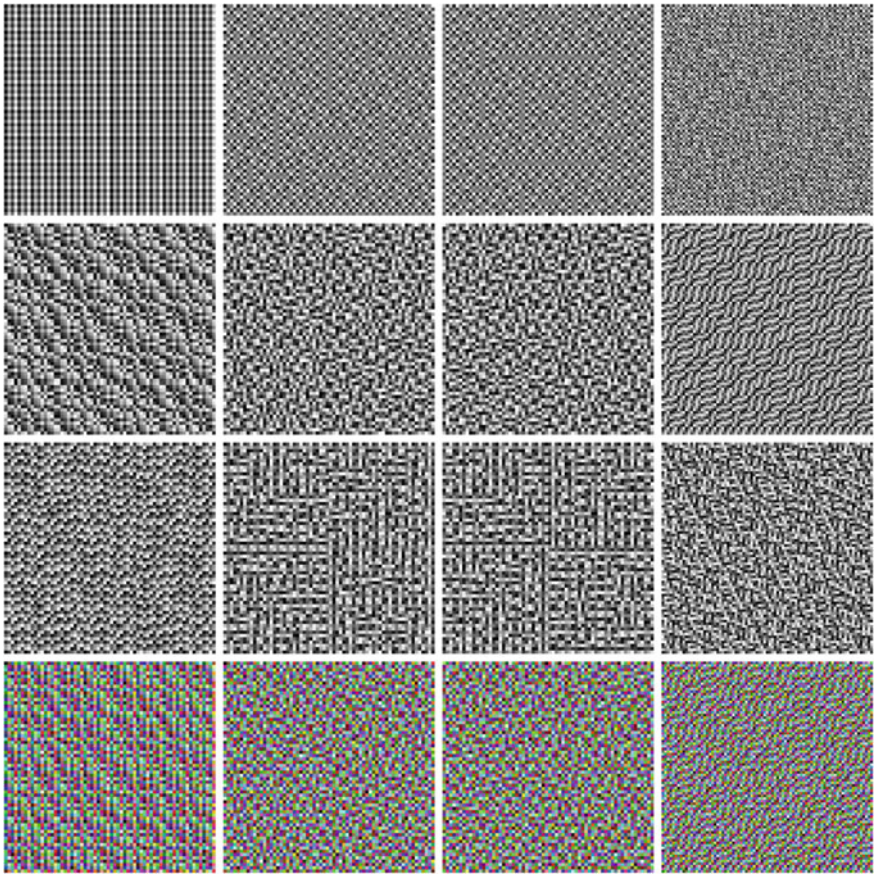


Fig. 3 From top to bottom, the rows display the values of the radical inverses ϕ_2, ϕ_3, and ϕ_5 as gray values, and all previous three superimposed by assigning them to the RGB channels of an image. The columns indicate the space filling curve used for enumeration. The resolution for the Morton, Hilbert, and Moore curves in base $b = 2$ is 64×64 pixels, while for the Peano curve in base $b = 3$ we display 81×81 pixels. For the eye it is easy to spot regular structures that are caused by correlations between the single low discrepancy sequences and the space filling curves. Superimposing them as in the bottom row, it becomes harder to identify the correlations with the Moore and Hilbert curves

are relatively co-prime. The uniformity of the simple construction can be improved by applying a permutation to the $k\text{-}th$ digit $a_k(i)$ of the index i represented in base b before radical inversion. Zaremba [31] has been successful with the simple permutation $\pi_b(a_k(i)) := (a_k(i) + k) \bmod b$, while later on Faure [8] developed a more general set of permutations improving upon Zaremba's results.

Now taking a look at the Morton curve in Fig. 2, it becomes obvious that the Morton index is either odd or even per column of pixels. As a consequence, computing the

radical inverse ϕ_2 of this index, which amounts to bits reversal of the index, results in $\phi_2 < \frac{1}{2}$ in even columns and $\phi_2 \geq \frac{1}{2}$ in odd columns [2, Sect. 3.1]. Similarly, the second least significant bit of the Morton index is on and off along the rows of pixels. This correlation results in striping artifacts along rows and columns of pixels, especially visible at one sample per pixel as shown in Fig. 3.

Other space-filling curves expose correlations, too, the reason being that the differences of indices of pixels are deterministic and correlated. Figure 2 shows the index differences larger than one for neighboring pixels. While the Hilbert and Moore curves are in base $b = 2$, they expose correlations with ϕ_3 and higher bases. For example, stripes are visible along lines of differences that are a multiple of the radical inverses' base 3. Similarly, the Hilbert curve has many adjacent pairs of pixels, whose difference of indices is 3. Radical inverses in these pixels in base $b = 3$ hence are correlated. The larger such clusters, the more prominent is the visible artifact.

While correlations are to be expected when the base of the low discrepancy sequence and the space filling curve are not co-prime, correlation structures may become visible whenever the differences of the pixel indices along a space filling curve are correlated to the base of the low discrepancy sequence in a regular way.

Owen scrambling [24] may resolve these correlations, because it recursively partitions the unit interval and randomly swaps the partitions independently. For the case of the Morton curve, Ahmed [2] developed a scrambling scheme, where it is sufficient to apply the recursive random swapping procedure to the index of the pixel along the Morton curve. The algorithm amounts to applying random permutations to the contiguous block of indices belonging to each quadrant along the hierarchy of the Morton curve.

As exemplified in Fig. 4, even recursive scrambling cannot remove all correlation artifacts. For example, in base $b = 2$, the recursive structure of scrambling correlates with the block structure of the Morton, Moore, and Hilbert curves. Yet, visible correlation artifacts are attenuated. Note that for the most significant bit both digit scrambling and Owen scrambling are identical. No matter how or whether this bit is scrambled, consecutive pixels along a space filling curve are hence larger and less or equal to $\frac{1}{2}$. In contrast to the Morton curve, this guarantees a good uniformity of values in the neighborhood of each pixel for the Hilbert, Moore, and Peano curves. For the example of base $b = 5$, scrambling may attenuate the visible structures and yet cannot resolve the correlations of the base of the radical inverse and the difference of pixel indices as shown in Fig. 2.

3.2 Blue-Noise Dithered Sampling

While some correlations are visible in low dimensions and at low sampling rates, sampling light transport paths requires many more dimensions. Figure 3 illustrates that already overlaying the first three radical inverses as RGB values hides most of the disturbing artifacts.

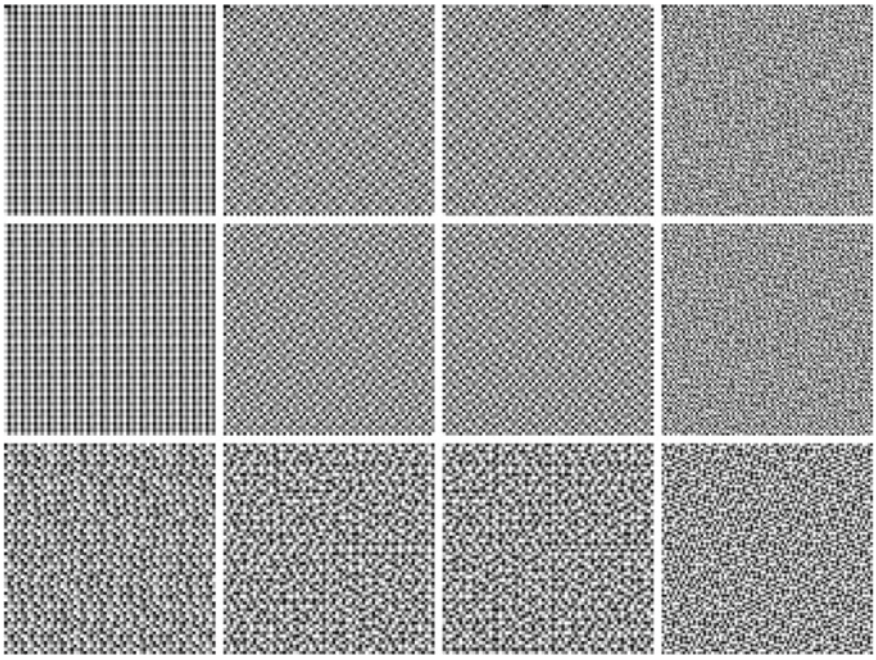


Fig. 4 While scrambling may attenuate visible correlations between pixels, it cannot completely remove the correlations. The first row shows ϕ_2 with random digit scrambling and the second row shows ϕ_2 with Owen scrambling. The bottom row shows ϕ_5 improved according to Faure [9]. As compared to Fig. 3, scrambling ϕ_2 does not at all help when used with the Morton curve and still leaves some visible lines indicative of the quadrant structure of the Hilbert and Moore curves. For the example of ϕ_5, scrambling helps most for the correlations when using the Peano curve but does not dramatically attenuate the artifacts when using the other space filling curves

The maps in Fig. 3 resemble the maps used for blue-noise dithered sampling [10] and may be used for the same purposes. As opposed to the optimization process required to create blue-noise dither maps, enumerating low discrepancy sequences along a space-filling curve allows one to approximate the desired spectral properties by just selecting components without the restriction to low dimension and without the need to store lookup tables. This approach is partially explored in Sect. 4.1.

4 Progressive Image Synthesis

Progressive image synthesis continues sampling within a given sample or time budget or terminates sampling once a selected image quality has been reached [19]. In what follows, we discuss extensions of the consistent algorithm of the previous section to enable adaptive sampling.

4.1 Deterministic Cranley-Patterson Rotation

Similar to the algorithm detailed in Sect. 3, a low discrepancy sequence is enumerated along the Hilbert curve at one sample per pixel. The vector of the low discrepancy sequence assigned to a pixel then is used to perform a deterministic Cranley-Patterson rotation [6]. This way, the same sequence of samples may be used across all pixels, however, shifted individually.

The Cranley-Patterson rotation is implemented as component-wise addition modulo one. Yet, using one and the same low discrepancy sequence for both shifting and sampling may expose visible correlation artifacts at low sampling rates. This is the case when enumerating the improved Halton sequence [9] along the Hilbert curve to shift the same sequence per pixel. While Cranley-Patterson rotations work with any point set, they work best with a point set designed for the unit torus such as rank-1 lattices and rank-1 lattice sequences [14]. For results, see Fig. 5.

4.2 Randomization

Array-RQMC algorithms [21] randomize the low discrepancy sequence for each iteration. This approach is straightforward to apply to the algorithm in Sect. 3: for each pass, the low discrepancy sequence is randomized and the results are accumulated until the termination by an empirical error criterion is triggered or a given time budget expires. The repeated randomization will eventually average out the correlation artifacts in the rendered frame.

While randomizing the low discrepancy sequence and accumulating results is unbiased and allows for unbiased variance estimation, some uniformity and hence convergence speed is sacrificed. Therefore, we aim at a deterministic and consistent algorithm, which in addition is simpler to execute and reproduce on massively parallel computer systems [15, 19].

4.3 Contiguous Segments of one Low Discrepancy Sequence

Progressive sampling may be implemented by iterating the algorithm in Sect. 3. To increase the uniformity of the samples in a pixel, one approach is to double the sampling rate with each iteration. Randomizing the low discrepancy sequence freshly for each iteration, the scheme is unbiased, see Sect. 4.2.

A deterministic variant of the algorithm sequentially consumes the points of the low discrepancy sequence along the space filling curve according to the selected number of samples per pixel along the iterations. The segment lengths drawn from the sequence then are a multiples of the length of the selected space filling curve. In

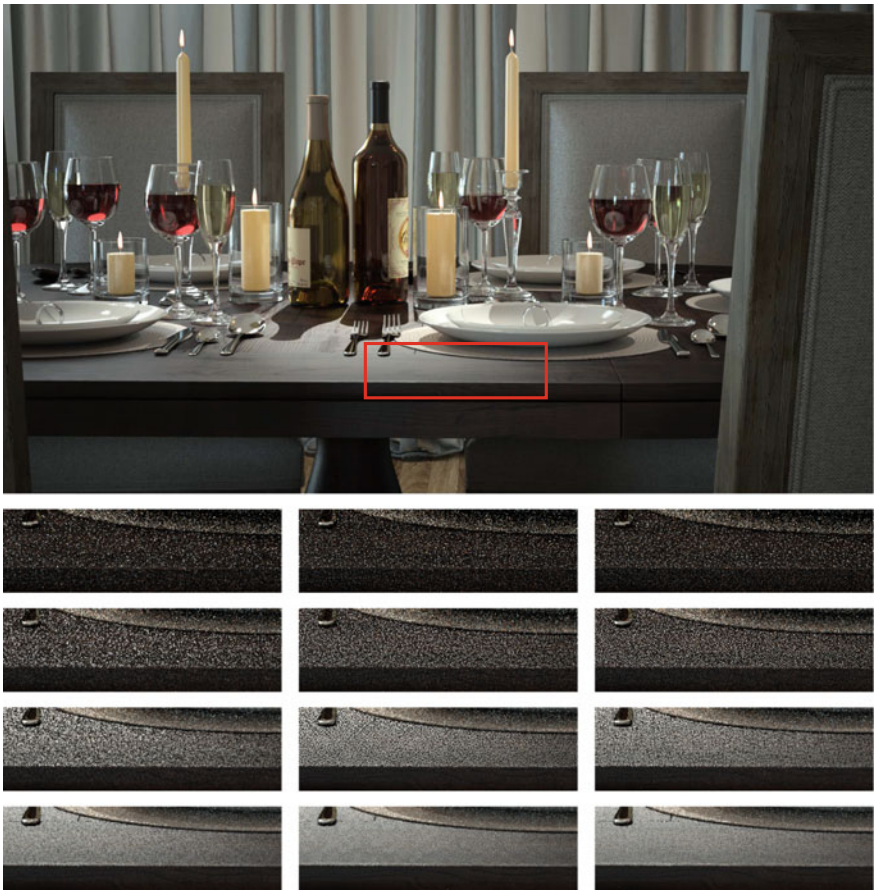


Fig. 5 Photorealistic image synthesis using one low discrepancy sequence across the image plane (left column, [12]), drawing samples from one low discrepancy sequence while enumerating the pixels along the Hilbert curve to shift an extensible lattice for sampling inside a pixel (middle column, Sect. 4.1) and in contiguous blocks to directly sample inside a pixels (right column, Sect. 4.4). The top image has been rendered using 100,000 samples per pixel, while the insets from top to bottom were rendered at 1, 4, 16, and 64 samples per pixel, respectively. The more uniformly distributed and less splotchy appearance of the sampling noise is especially visible in areas that at higher sampling rates appear smooth, like the table top or the back rests. The difference in quality is clearly visible on a computer screen and may be difficult to reproduce in print. The reader may need to vary the distance of observation. As all methods are consistent, the observable differences vanish with an increasing number of samples per pixel. Nevertheless, the improvement very much matters in settings, where only a small number of samples are affordable, such as in real-time rendering

two dimensions, the length is a quadratic power of the base of the selected space-filling curve.

For the example of base two, segment lengths amount to multiples of powers of two. Hence, for a specific pixel, samples drawn from a radical inverse in base $b = 2$ are spaced in multiples of powers of two, which will reveal correlation artifacts as visualized in Fig. 2. Depending on the combination of space filling curve and component of the low discrepancy sequence, samples may even not be distributed uniformly, as explained in Sect. 3.1. Omitting the components of the low discrepancy sequence that correlate with the space-filling curve may help, but as revealed in Sect. 3.1, there may be multiple components with correlations.

The issues of leapfrogging low discrepancy sequences have been encountered in parallelizing quasi-Monte Carlo methods [1, 20]. Their remedy has led to the concept of partitioning low discrepancy sequences into multiple low discrepancy sequences, which we explore next.

4.4 Partitioning one Low Discrepancy Sequence

Partitioning a low discrepancy sequence into a finite number of low discrepancy sequences has been introduced in [18] for the purpose of parallel quasi-Monte Carlo methods: One dimension of a low discrepancy sequence is used for partitioning, while the remaining dimensions are used for quasi-Monte Carlo integration. We use the principle to develop a simple consistent algorithm for rendering along the Hilbert curve.

Let $\phi_b(i)$ be the component of a low discrepancy sequence to be partitioned into $N = b^m$ low discrepancy sequences and let $\mathbf{x}_i$ be the points of that low discrepancy sequence without the component used for partitioning. Then the integers

$$\lfloor N \cdot \phi_b(i) \rfloor = \lfloor b^m \cdot \phi_b(i) \rfloor = \left\lfloor b^m \cdot \sum_{k=0}^{\infty} a_k(i) b^{-k-1} \right\rfloor,$$

form a permutation of $\{0, \ldots, N-1\}$ that repeats every N points. Selecting N as the length of a space filling curve and b its base, each pixel with index j along the space-filling curve is assigned the sequence of points

$$P_j = \left\{ \mathbf{x}_{l \cdot N + \phi_b^{-1}(j/N)} : l \in \mathbb{N}_0 \right\} \Leftrightarrow P_{\phi_b^{-1}(j/N)} = \left\{ \mathbf{x}_{l \cdot N + j} : l \in \mathbb{N}_0 \right\}$$

of the original low discrepancy sequence $\mathbf{x}_i$, which results in an overall consistent deterministic quasi-Monte Carlo method [15, Sect. 1.1].

As the offset $\phi_b^{-1}(j/N)$ is constant per pixel, omitting the inverse of the permutation $\lfloor b^m \cdot \phi_b(i) \rfloor$ and instead assigning

$$P_j = \left\{ \mathbf{x}_{l \cdot N + j} : l \in \mathbb{N}_0 \right\}$$

simplifies the implementation: Now contiguous blocks of the low discrepancy sequence are enumerated along the space-filling curve, and we have the additional benefit of locally improved uniformity as described in Sect. 3 and illustrated in Fig. 1.

Combining the Hilbert curve and the improved Halton sequence [9] for progressive image synthesis, we have $b = 2$ and use ϕ_2 to partition the low discrepancy sequence into $N = 2^{2n}$ low discrepancy sequences, one for each pixel along the Hilbert curve. The algorithm uses the pixel index j along the Hilbert curve as offset into the low discrepancy sequence $\mathbf{x}_i$ and leapfrogs from there with a stride of N, which is simple to execute on a massively parallel computer system, like for example a GPU cluster [3, 18, 19].

5 Results and Discussion

The Hilbert curve has been applied in computer graphics before. Rendering images by enumerating pixels along the Hilbert curve improves performance by higher cache hit rates due to the locality properties of the Hilbert curve. The visual benefits of half-toning by dithering along the Hilbert curve have been recognized in [28].

Our new algorithms benefit from these findings. They are a special case of Array-RQMC that does not require sorting because we rely on the bijection between pixels and the space-filling curves. As indices can be computed directly, neither lookup tables nor additional memory for lookup tables are required.

Our focus is on deterministic algorithms, as these can be reliably parallelized and results are exactly reproducible [15]. We use an improved variant of the Halton sequence [9] in the experiments. The implementation of fitting elementary intervals to the pixel raster [12] is involved and requires 64-bit signed integers to run Euclid's algorithm for the Chinese remainder theorem. This computation is not required for the algorithms in Sects. 3 and 4.4 that are straightforward to implement. Shifting a rank-1 lattice sequence [14] by the Halton sequence enumerated along the Hilbert curve (see Sect. 4.1) is even simpler and practical with only two 32-bit integer indices. We employ an extensible lattice constructed by primitive polynomials [16]. To assess the visual differences of the classic [12] and the two new progressive sampling approaches, their results are compared at low sampling rates in Fig. 5.

The most prominent advantage of the new algorithms is inherited from Array-RQMC: As illustrated in Fig. 1, the Hilbert curve has the smallest number of curve segments in the neighborhood around a pixel. Hence more consecutive samples are used locally, which makes better use of the uniformity of a low discrepancy sequence across pixels as compared to other space-filling curves. Hence, the noise in the images is more uniformly distributed noise at low sampling rates.

Our methods achieve a quality comparable to methods that require optimization [10, 13], are available for any number of dimensions, are simpler than other approaches that sample along space-filling curves [2], are deterministic, and are consistent.

We have not yet explored the potential of selecting or reordering the dimensions of low discrepancy sequences. This is an interesting direction of future research that has been initially explored for rank-1 lattices in computer graphics [22]. Furthermore, it is worth investigating the many other possible combinations of low discrepancy sequences and space-filling curves with respect to their visual quality and convergence speed.

6 Conclusion

Based on the seminal work on Array-RQMC [21], we introduced simple deterministic consistent rendering algorithms that at low sampling rates produce noise characteristics that are very amenable to the human eye. Key to the algorithms are the preservation of discrepancy when enumerating low discrepancy sequences along the Hilbert curve and the principle of partitioning one low discrepancy sequence into multiple. It appears that the correlation of samples across pixels via low discrepancy may be more relevant to the eye than their independence.

References

1. Abramov, G.: US patent #6,911,976: System and method for rendering images using a strictly-deterministic methodology for generating a coarse sequence of sample points (2002). Assignee: mental images GmbH. Berlin, DE
2. Ahmed, A.G.M., Wonka, P.: Screen-space blue-noise diffusion of Monte Carlo sampling error via hierarchical ordering of pixels. ACM Trans. Graph. **39**(6) (2020). https://doi.org/10.1145/3414685.3417881
3. van Antwerpen, D., Seibert, D., Keller, A.: A simple load-balancing scheme with high scaling efficiency. In: E. Haines, T. Akenine-Möller (eds.) Ray Tracing Gems. Apress (2019). http://raytracinggems.com
4. Binder, N., Fricke, S., Keller, A.: Massively parallel path space filtering. In: Keller, A. (ed.) Monte Carlo and Quasi-Monte Carlo Methods 2020. Springer (2022). http://arxiv.org/abs/1902.05942
5. Cook, R.L.: Stochastic sampling in computer graphics. ACM Trans. Graph. **5**, 51–72 (1986)
6. Cranley, R., Patterson, T.: Randomization of number theoretic methods for multiple integration. SIAM J. Numer. Anal. **13**, 904–914 (1976)
7. Ernst, M., Stamminger, M., Greiner, G.: Filter importance sampling. In: Proceedings of 2006 IEEE/EG Symposium on Interactive Ray Tracing, pp. 125–132 (2006)
8. Faure, H.: Good permutations for extreme discrepancy. J. Number Theory **42**, 47–56 (1992)
9. Faure, H., Lemieux, C.: Generalized Halton sequences in 2008: A comparative study. ACM Trans. Model. Comp. Simul. **19**(4), 15:1–15:31 (2009)
10. Georgiev, I., Fajardo, M.: Blue-noise dithered sampling. ACM SIGGRAPH 2016 Talks (2016)
11. Gerber, M., Chopin, N.: Sequential quasi Monte Carlo. J. Roy. Stat. Soc. Ser. B (Statistical Methodology) **77**(3), 509–579 (2015). http://www.jstor.org/stable/24774819
12. Grünschloß, L., Raab, M., Keller, A.: Enumerating quasi-Monte Carlo point sequences in elementary intervals. In: Plaskota, L., Woźniakowski, H. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2010, pp. 399–408. Springer (2012). http://gruenschloss.org/sample-enum/sample-enum.pdf

13. Heitz, E., Belcour, L., Ostromoukhov, V., Coeurjolly, D., Iehl, J.C.: A low-discrepancy sampler that distributes Monte Carlo errors as a blue noise in screen space. In: SIGGRAPH'19 Talks. ACM, Los Angeles, United States (2019). https://hal.archives-ouvertes.fr/hal-02150657
14. Hickernell, F., Hong, H., L'Ecuyer, P., Lemieux, C.: Extensible lattice sequences for quasi-Monte Carlo quadrature. SIAM J. Sci. Comput. **22**, 1117–1138 (2001)
15. Keller, A.: Quasi-Monte Carlo image synthesis in a nutshell. In: Dick, J., Kuo, F., Peters, G., Sloan, I. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2012, pp. 203–238. Springer (2013)
16. Keller, A., Binder, N., Wächter, C.: Construction of a rank-1 lattice sequence based on primitive polynomials. In: Larcher, G., Pillichshammer, F., Winterhof, A., Xing, C. (eds.) Applied Algebra and Number Theory, pp. 204–215. Cambridge University Press (2014). 10.1017/CBO9781139696456.013
17. Keller, A., Georgiev, I., Ahmed, A., Christensen, P., Pharr, M.: My favorite samples. In: ACM SIGGRAPH 2019 Courses, SIGGRAPH '19, pp. 15:1–15:271. ACM, New York, NY, USA (2019). https://doi.org/10.1145/3305366.3329901
18. Keller, A., Grünschloß, L.: Parallel quasi-Monte Carlo integration by partitioning low discrepancy sequences. In: Plaskota, L., Woźniakowski, H. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2010, pp. 487–498. Springer (2012). http://gruenschloss.org/parqmc/parqmc.pdf
19. Keller, A., Wächter, C., Raab, M., Seibert, D., Antwerpen, D., Korndörfer, J., Kettner, L.: The Iray light transport simulation and rendering system (2017). CoRR abs/ arXiv:1705.01263
20. Kocis, L., Whiten, W.: Computational investigations of low-discrepancy sequences. ACM Trans. Math. Softw. **23**(2), 266–294 (1997). http://doi.acm.org/10.1145/264029.264064
21. L'Ecuyer, P., Munger, D., Lécot, C., Tuffin, B.: Sorting methods and convergence rates for Array-RQMC: Some empirical comparisons. In: Mathematics and Computers in Simulation, vol. 143 (2018)
22. Liu, H., Han, H., Jiang, M.: Rank-1 lattices for efficient path integral estimation. Comput. Graph. Forum **40**(2), 91–102 (2021)
23. Niederreiter, H.: Random Number Generation and Quasi-Monte Carlo Methods. SIAM, Philadelphia (1992)
24. Owen, A.: Randomly permuted (t, m, s)-nets and (t, s)-sequences. In: Niederreiter, H., Shiue, P. (eds.) Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing. Lecture Notes in Statistics, vol. 106, pp. 299–315. Springer (1995)
25. Paulin, L., Coeurjolly, D., Bonneel, N., Iehl, J.C., Keller, A., Ostromoukhov, V.: Matbuilder: Mastering sampling uniformity over projections. ACM Trans. Graph. **41**(4), 84:1–84:13 (2022)
26. Paulin, L., Coeurjolly, D., Iehl, J.C., Bonneel, N., Keller, A., Ostromoukhov, V.: Cascaded Sobol' sampling. ACM Trans. Graph. **40**(6), 274:1–274:13 (2021). https://hal.archives-ouvertes.fr/hal-03358957
27. Pharr, M., Jacob, W., Humphreys, G.: Physically Based Rendering - From Theory to Implementation, 3rd edn. Morgan Kaufmann (2016)
28. Velho, L., Gomes, J.d.M.: Digital halftoning with space filling curves. In: Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '91, p. 81–90. Association for Computing Machinery, New York, NY, USA (1991). https://doi.org/10.1145/122718.122727
29. Wächter, C., Keller, A.: Efficient simultaneous simulation of Markov chains. In: Keller, A., Heinrich, S., Niederreiter, H. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2006, pp. 669–684. Springer (2007)
30. Yellot, J.: Spectral consequences of photoreceptor sampling in the rhesus retina. Science **221**, 382–385 (1983)
31. Zaremba, S.: La discrépance isotrope et l'intégration numérique. Ann. Mat. Pura Appl. **87**, 125–136 (1970)

Array-RQMC to Speed up the Simulation for Estimating the Hitting-Time Distribution to a Rare Set of a Regenerative System

Marvin K. Nakayama and Bruno Tuffin

Abstract Estimating the distribution of the hitting time to a rarely visited set of states presents substantial challenges. We recently designed simulation-based estimators to exploit existing theory for regenerative systems that a scaled geometric sum of independent and identically distributed random variables weakly converges to an exponential random variable as the geometric's parameter vanishes. The resulting approximation then reduces the estimation of the distribution to estimating just the mean of the limiting exponential variable. The present work examines how randomized quasi-Monte Carlo (RQMC) techniques can help to reduce the variance of the estimators. Estimating hitting-time properties entails simulating a stochastic (here Markov) process, for which the so-called array-RQMC method is suited. After describing its application, we illustrate numerically the gain on a standard rare-event problem. This chapter combines ideas from several areas in which Pierre L'Ecuyer has made fundamental theoretical and methodological contributions: randomized quasi-Monte Carlo methods, rare-event simulation, and distribution estimation.

Keywords Rare event simulation · Distribution estimation · Randomized quasi-Monte Carlo

1 Introduction

Monte Carlo (MC) simulation provides a primary tool to estimate the probability of rare events or related indicators [27]. The extensive related literature focuses mainly on estimating the *mean* of a relevant random variable, but its *distribution* provides valuable additional information. For example, suppose a manufacturer wants to specify an appropriate length of a warranty. While the product's mean time to failure

M. K. Nakayama
Department of Computer Science, New Jersey Institute of Technology, Newark, NJ 07102, USA
e-mail: marvin@njit.edu

B. Tuffin (✉)
Inria, University Rennes, CNRS, IRISA, Campus de Beaulieu, 35042 Rennes, France
e-mail: bruno.tuffin@inria.fr

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_17

(MTTF) yields some relevant details, more useful are the random failure time's quantiles (i.e., inverse distribution). Setting the warranty length to, say, the 0.9-quantile leads to 10% of products resulting in warranty claims.

Distribution determination of a hitting time, especially related to a rare event (e.g., system failure), poses numerous challenges. But when the simulated stochastic process is *regenerative* [10], existing theory [11] shows that the hitting time to a rare set converges weakly to an exponential random variable if the probability to hit the rare set before regenerating converges to zero. This suggests approximating the hitting-time distribution by an exponential, reducing distribution estimation to estimating just its mean, which is broadly covered in the rare-event simulation literature. Our papers [4, 5] present two MC estimators exploiting such approximations. The *exponential estimator* directly applies this idea, further employing *measure-specific importance sampling* (MSIS) [6] to efficiently estimate the mean. The other is the *convolution estimator*, which first applies an exponential approximation to the distribution of the geometric sum of cycle lengths (i.e., the times elapsing between successive regenerations) completed before the first visit to the rare set, and then convolves this with the distribution of the hitting time given that it occurs in a cycle.

This chapter investigates how randomized quasi-Monte Carlo (RQMC) can be used to improve the accuracy of the above estimators and the potential associated gains. By distributing the sample points more evenly than independent sampling on the considered domain, RQMC methods can reduce the variance of estimators and even increase the convergence speed to the true value [13]. A naive implementation of RQMC to simulate a stochastic process entails generating sequences whose dimension is at least the number of transitions in a simulated path, which is typically large or even unbounded. But RQMC often performs poorly in large or infinite dimensions. Array-RQMC [16, 17, 20] has been designed precisely to simulate Markov chains while retaining the power of RQMC, the dimension of the generated sequences being "just" the required number of random values to simulate a single step of the chain. Basically, array-RQMC simulates in parallel a set of realizations of a Markov chain and makes use of a "sorting function" to reorder the chains according to their states after each simulation step. We describe the array-RQMC implementations of the exponential and convolution estimators and illustrate numerically the gains that can be derived from it.

Interestingly, this work combines several research interests of Pierre L'Ecuyer: rare-event simulation [14, 15]; RQMC techniques [13], among which array-RQMC [16, 17, 20] is specifically designed by Pierre and his coauthors to simulate Markov chains; and distribution determination [1, 21].

The remainder of this chapter unfolds as follows. Section 2 reviews the exponential and convolution estimators devised in [4]. Section 3 recalls array-RQMC simulation methods and how it can be implemented for our problem. As an illustration, we apply the approach in Sect. 4 to a standard rare-event problem in the literature: the hitting time to a large buffer threshold in an M/M/1 queue. Finally, Sect. 5 concludes the paper and provides further research directions to pursue on these ideas.

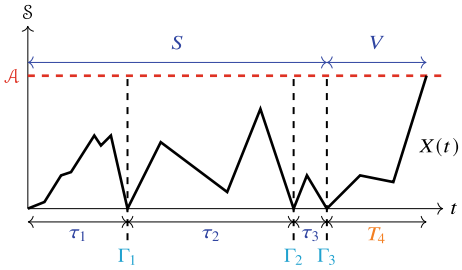


Fig. 1 Illustration of the notation used to represent and analyze regenerative processes, where $M = 3$

2 Regenerative-Simulation-Based Estimators of the Distribution of the Hitting Time to a Rarely Visited Set

2.1 Assumptions and Notations

For $\mathbb{R}_+$ denoting the set of nonnegative real numbers, we consider, for ease of exposition, a positive recurrent Markov chain $(X(t) : t \in \mathbb{R}_+)$ defined on a discrete state space $\mathcal{S}$. Our goal is to estimate the cumulative distribution function (cdf) F of the hitting time $T = \inf\{t \geq 0 : X(t) \in \mathcal{A}\}$ of a subset $\mathcal{A}$ of $\mathcal{S}$, as well as the q-quantile $\xi = \xi_q = F^{-1}(q) = \inf\{t : F(t) \geq q\}$ of F (or of T) for some $q \in (0, 1)$.

Define regeneration times $0 = \Gamma_0 < \Gamma_1 < \cdots$ (always existing with our assumptions of discrete $\mathcal{S}$ and recurrence: it suffices to consider return times to a fixed state as regeneration times) and $\tau_k = \Gamma_k - \Gamma_{k-1}$, the length between regeneration $k-1$ and regeneration k for $k \geq 1$. The process "probabilistically restarts" at each regeneration time Γ_k. The process between successive regenerations is called a *cycle*, the k-th cycle being $(X(\Gamma_{k-1} + s) : 0 \leq s < \tau_k)$. The couples $(\tau_k, (X(\Gamma_{k-1} + s) : 0 \leq s < \tau_k) : k \geq 1)$ are independent and identically distributed (i.i.d.), and let τ denote a generic copy of τ_k. Let $T_k = \inf\{t \geq 0 : X(\Gamma_{k-1} + t) \in \mathcal{A}\}$ be the first hitting time to $\mathcal{A}$ after regeneration time Γ_{k-1}. We further define $M = \inf\{i \geq 1 : T_i < \tau_i\} - 1$ as the number of cycles completed before first hitting $\mathcal{A}$. As the cycles are i.i.d., M obeys a geometric distribution with parameter $p = \mathbb{P}(T < \tau)$ and support starting from 0; i.e., $\mathbb{P}(M = k) = (1-p)^k p$ for each $k \in \{0, 1, 2, \ldots\}$.

We can express

$$T = S + V \equiv \sum_{i=1}^{M} \tau_i + T_{M+1}, \tag{1}$$

where the regenerative property ensures the geometric sum $S = \sum_{i=1}^{M} \tau_i$ is independent of $V = T_{M+1}$. Define G as the cdf of S, and H the cdf of V. Note that H is the conditional cdf of T_1, given $T_1 < \tau_1$. Figure 1 illustrates the notation, with the state space $\mathcal{S}$ on the vertical axis and $\mathcal{A}$ the subset above the horizontal dashed line.

2.2 Exponential Limit

We consider a rare-event setting where the probability p to reach $\mathcal{A}$ before regeneration is small. To examine the asymptotic properties of estimators as the probability shrinks, we index the model and all notation by a rarity parameter $\epsilon > 0$, such that $p \equiv p_\epsilon \to 0$ as $\epsilon \to 0$. But when unambiguous, we will omit the index ϵ to simplify notation. Two well-known rare-event contexts having $p_\epsilon \to 0$ as $\epsilon \to 0$ are the following [7]:

- Stable queueing system: For a single-server queue with first-in-first-out discipline, we want to estimate the distribution of the hitting time T to a large buffer size buffer size $b \equiv b_\epsilon = \lceil 1/\epsilon \rceil$. Specifically, $X(t)$ denotes the total number of customers in the system at time $t \geq 0$, and the state space is $\mathcal{S} = \{0, 1, 2, \ldots\}$. Thus, the hitting time is $T = \inf\{t \geq 0 : X(t) \in \mathcal{A}\}$ with $\mathcal{A} = \mathcal{A}_\epsilon = \{b_\epsilon, b_\epsilon + 1, \ldots\}$. For a G/G/1 queue, regenerations occur when a customer arrives to an empty system, which we assume occurs at time $t = 0$. In our numerical illustrations with an M/M/1 queue, returns to any fixed state constitutes a regeneration sequence, where we take the fixed state to be 0 and $X(0) = 0$. The transition kernel does not depend on ϵ, and rarity arises from b_ϵ being large for small ϵ, and [28] shows that $p_\epsilon \to 0$ as $\epsilon \to 0$.
- Highly reliable Markovian system (HRMS) considered in dependability analysis: The system consists of components of different types, each having a specified redundancy. Each component is subject to failures and repairs, all being exponentially distributed with rates depending on the component type. Failure propagations can occur, i.e., a component failure can cause others to simultaneously fail. A state $x \in \mathcal{S}$ specifies the number of components failed of each type, as well as any other necessary information (e.g., about queueing of failed components waiting for repair) so that the resulting stochastic process on state space $\mathcal{S}$ is a Markov chain. The entire system is considered down (i.e., in $\mathcal{A}$) when specified combinations of components are currently failed. We may want to estimate the distribution of the hitting time to $\mathcal{A}$ when all components are operational at time $t = 0$. Rarity comes from failure rates being small (depending on ϵ) with respect to repair rates, leading to probabilistically long hitting times, and [29] provides conditions ensuring that $p_\epsilon \to 0$ as $\epsilon \to 0$.

Let $\mu_\epsilon = \mathbb{E}_\epsilon[T_\epsilon]$ be the mean hitting time, with $\mu_\epsilon \to \infty$ as $\epsilon \to 0$. Then existing limit results (see [10, 11]) show that if $p_\epsilon \to 0$ as $\epsilon \to 0$, then the normalized random variable T_ϵ/μ_ϵ converges weakly to an exponential, i.e., for $t \in \mathbb{R}$ and $t^+ = \max(t, 0)$,

$$\lim_{\epsilon \to 0} \mathbb{P}_\epsilon(T_\epsilon/\mu_\epsilon \leq t) = 1 - e^{-t^+}. \tag{2}$$

2.3 Exponential Estimators with Monte Carlo (MC)

From the limiting behavior (2), we can write for fixed small $\epsilon > 0$

$$F_\epsilon(t) = \mathbb{P}_\epsilon(T_\epsilon < t) = \mathbb{P}_\epsilon(T_\epsilon/\mu_\epsilon < t/\mu_\epsilon) \approx 1 - e^{-t^+/\mu_\epsilon} \equiv \widetilde{F}_\epsilon(t). \tag{3}$$

Thus to compute the cdf of hitting time T (dropping now the subscript ϵ to ease notation, as we will often but not always do in the following), we asymptotically need to know "just" its mean. (This is analogous to the central limit theorem (CLT), where the asymptotic normal cdf is fully specified through simply its mean and variance). Estimating the mean μ has been extensively studied in the literature. Doing this by averaging i.i.d. copies of T can be time-consuming because generating each observation of T typically entails lengthy simulations (e.g., many transitions) for small ϵ. Instead, we exploit the regenerative structure to rewrite μ as (see [6])

$$\mu = \frac{\mathbb{E}[T \wedge \tau]}{\mathbb{P}(T < \tau)} \equiv \frac{\zeta}{p}, \tag{4}$$

where $x \wedge y = \min(x, y)$. The key point is that (4) expresses μ in terms of cycle-based quantities, ζ and p, each of which can be estimated by simulating only cycles. The numerator $\zeta = \mathbb{E}[T \wedge \tau]$ in (4) can usually be estimated well by crude Monte Carlo, while the denominator $p = \mathbb{P}(T < \tau)$ is a small probability for which rare-event simulation techniques have to be applied for efficient estimation. Measure-specific importance sampling [6] employs independent simulations to estimate the numerator and denominator using crude simulation (CS) and importance sampling (IS), respectively. Given a computation budget of simulating n cycles in total to estimate μ, MSIS allocates a proportion $\gamma \in (0, 1)$ (resp., $1 - \gamma$) of the budget for CS (resp., IS). More specifically,

- We use $n_{\mathrm{CS}} \equiv \gamma n$ cycles to estimate the numerator ζ in (4) by CS via

$$\widehat{\zeta}_n = \frac{1}{n_{\mathrm{CS}}} \sum_{i=1}^{n_{\mathrm{CS}}} T_i \wedge \tau_i \tag{5}$$

  from n_{CS} independent observations $T_i \wedge \tau_i$ $(1 \le i \le n_{\mathrm{CS}})$ generated using the original system dynamics, denoted by $\mathbb{P}$.
- Because CS is unlikely to observe the event $T < \tau$ when p is small, MSIS instead estimates the denominator p in (4) using $n_{\mathrm{IS}} \equiv (1 - \gamma)n$ cycles generated using IS. IS entails simulating under another probability measure $\mathbb{P}'$ rather than the original measure $\mathbb{P}$, where $\mathbb{P}'$ is chosen so that $T < \tau$ is more likely and can depend on ϵ. Letting $\mathcal{I}(\cdot)$ be the indicator function, we apply a "change of measure" to write

$$p = \mathbb{E}[\mathcal{I}(T < \tau)] = \int \mathcal{I}(T < \tau)\, \mathrm{d}\mathbb{P} = \int \mathcal{I}(T < \tau) L\, \mathrm{d}\mathbb{P}' = \mathbb{E}'[\mathcal{I}(T < \tau)L],$$

  with $L = \mathrm{d}\mathbb{P}/\mathrm{d}\mathbb{P}'$ the likelihood ratio, and $\mathbb{E}'$ denotes expectation under measure $\mathbb{P}'$. An unbiased estimator of p is then

$$\widehat{p}_n = \frac{1}{n_{\rm IS}} \sum_{i=1}^{n_{\rm IS}} \mathcal{I}(T_i' < \tau_i') L_i', \tag{6}$$

for i.i.d. copies $(\mathcal{I}(T_i' < \tau_i'), L_i')$, $i = 1, 2, \ldots, n_{\rm IS}$, of $(\mathcal{I}(T < \tau), L)$ under $\mathbb{P}'$.

The resulting MSIS estimator of the mean μ in (4) is the ratio estimator

$$\widehat{\mu}_n = \frac{\widehat{\zeta}_n}{\widehat{p}_n}. \tag{7}$$

The proportion γ can be selected during a presimulation run to minimize the variance per unit of computational budget of $\widehat{\mu}_n$ (see [6] for details). To summarize [4]:

Definition 1 The *exponential estimator* of the cdf $F(t)$ of T is

$$\widehat{F}_{{\rm exp},n}(t) = 1 - e^{-t^+/\widehat{\mu}_n}. \tag{8}$$

For fixed $q \in (0, 1)$, the exponential estimator of the q-quantile $\xi = F^{-1}(q)$ is $\widehat{\xi}_{{\rm exp},n} = \widehat{F}^{-1}_{{\rm exp},n}(q) = -\widehat{\mu}_n \ln(1 - q)$.

As a notational convention, for an unknown parameter (e.g., F), we use a tilde to signify a non-simulation approximation (e.g., $\widetilde{F}_\epsilon$ in (3)) based on a weak-convergence result, as in (2). A hatted variable (e.g., $\widehat{F}_{{\rm exp},n}$) denotes a simulation estimator.

The exponential estimators in Definition 1 result from approximating the true cdf F by $\widetilde{F}_\epsilon$ in (3), with (2) showing that the approximation becomes exact as the rarity parameter $\epsilon \to 0$. But any actual system has a small but *fixed* $\epsilon > 0$, which typically leads to $\widetilde{F}_\epsilon \neq F$. Because the exponential estimators are estimating quantities related to the approximation $\widetilde{F}_\epsilon$ and not the actual F, the estimators have bias that does not vanish as the computing budget $n \to \infty$. For example, for fixed $\epsilon > 0$ and $t > 0$, we have that as $n \to \infty$, $\widehat{F}_{{\rm exp},n}(t) \equiv \widehat{F}_{{\rm exp},n,\epsilon}(t)$ converges almost surely to $\widetilde{F}_\epsilon(t) = 1 - e^{-t/\mu_\epsilon}$, not to $F(t)$.

For fixed $\epsilon > 0$, the exponential estimators obey CLTs as $n \to \infty$, but the CLTs will employ centering constants computed from $\widetilde{F}_\epsilon$ rather than F. For example, for fixed $\epsilon > 0$ and $t > 0$, the exponential cdf estimator satisfies $\sqrt{n}[\widehat{F}_{{\rm exp},n}(t) - \widetilde{F}_\epsilon(t)] \Rightarrow \mathcal{N}(0, \psi_t^2)$ as $n \to \infty$ for an asymptotic variance $\psi_t^2 \equiv \psi_{t,\epsilon}^2$ that can be derived using the delta method, where $\Rightarrow$ denotes weak convergence and $\mathcal{N}(a, b^2)$ is a normal random variable with mean a and variance b^2. Similarly, for fixed $\epsilon > 0$ and $q \in (0, 1)$, the exponential q-quantile estimator also obeys a CLT (as $n \to \infty$) with centering constant $\widetilde{\xi}_\epsilon \equiv -\mu_\epsilon \ln(1 - q)$ rather than the true q-quantile $\xi = F^{-1}(q)$. Based on these two CLTs, we can then provide confidence intervals (CIs) for the true values $F(t)$ and ξ, but the CIs are biased from fixing $\epsilon > 0$, so the coverage probabilities will converge to 0 as $n \to \infty$. A CI may still have reasonable coverage when the estimator's bias makes a negligible contribution to its mean square error. This may be difficult to determine in practice, as quantifying the bias is nontrivial, but may occur for large (but not too large) n and fixed small $\epsilon > 0$.

2.4 Convolution Estimators with Monte Carlo

Rather than directly approximating the cdf F by an exponential, as done for the exponential estimator, we instead can use the decomposition $T = S + V$ in Eq. (1), leading to expressing the cdf F as the convolution

$$F = G \star H, \text{ recalling that } S \sim G \text{ and } V \sim H \text{ are independent,} \tag{9}$$

where $(G \star H)(t) = \int H(t-y)\,\mathrm{d}G(y)$. Typically, the exponential limit (2) for the scaled hitting time arises from S (scaled by its mean $\eta = \mathbb{E}[S] = \mathbb{E}[M] \cdot \mathbb{E}[\tau \mid \tau < T]$) converging weakly to an exponential as $\epsilon \to 0$; e.g., see [11, Theorem 3.2.5]. Thus, for small $\epsilon > 0$, we approximate $G(y)$ by $\widetilde{G}_{\exp}(y) \equiv 1 - e^{-y^+/\eta}$. As before with the exponential estimator in (8), the approximation reduces estimation of the cdf G to estimating just its mean η. Writing $\mathbb{E}[M] = (1-p)/p$ and $\mathbb{E}[\tau \mid \tau < T] = \mathbb{E}[\tau \mathcal{I}(\tau < T)]/(1-p)$ suggests estimating $\eta = (1/p)\mathbb{E}[\tau \mathcal{I}(\tau < T)]$ by

$$\widehat{\eta}_n = \frac{1}{\widehat{p}_n\, n_{\mathrm{CS}}} \sum_{i=1}^{n_{\mathrm{CS}}} \tau_i \mathcal{I}(\tau_i < T_i),$$

where we can employ the same CS and IS cycle data from (5) and (6) used for the exponential estimator. This then yields the MSIS estimator of G in (9) as

$$\widehat{G}_{\exp,n}(t) = 1 - e^{-t/\widehat{\eta}_n}. \tag{10}$$

Estimating the cdf H of V in (9) also requires rare-event simulation techniques. As $H(x) = \mathbb{P}(T \le x \mid T < \tau) = \mathbb{P}(T \le x, T < \tau)/p$, a change of measure gives

$$H(x) = \frac{1}{p}\mathbb{E}[\mathcal{I}(T \wedge \tau \le x,\ T < \tau)] = \frac{1}{p}\mathbb{E}'[\mathcal{I}(T \wedge \tau \le x,\ T < \tau)\, L].$$

Applying IS produces a sample $(T_i' \wedge \tau_i', \mathcal{I}(T_i' < \tau_i'), L_i')$, $i = 1, 2, \ldots, n_{\mathrm{IS}}$, of $(T \wedge \tau, \mathcal{I}(T < \tau), L)$ under $\mathbb{P}'$ from n_{IS} cycles (as for the exponential estimator), leading to

$$\widehat{H}_n(x) = \frac{1}{\widehat{p}_n\, n_{\mathrm{IS}}} \sum_{i=1}^{n_{\mathrm{IS}}} \mathcal{I}(T_i' \wedge \tau_i' \le x, T_i' < \tau_i') L_i' \tag{11}$$

as an estimator of H. Convolving the two distributions $\widehat{G}_{\exp,n}$ from (10) and $\widehat{H}_n$ from (11), [4] obtains the following estimator of cdf F in (9).

Definition 2 The *convolution estimator* of the cdf $F(t)$ is

$$\widehat{F}_{\mathrm{conv},n}(t) = (\widehat{G}_{\exp,n} \star \widehat{H}_n)(t) = 1 - \frac{1}{\widehat{p}_n \cdot n_{\mathrm{IS}}} \sum_{i=1}^{n_{\mathrm{IS}}} \mathcal{I}(T_i' < \tau_i')\, L_i'\, e^{-(t-(T_i' \wedge \tau_i'))^+/\widehat{\eta}_n}.$$

The convolution estimator of the q-quantile $\xi = F^{-1}(q)$ is $\widehat{\xi}_{\text{conv},n} = \widehat{F}^{-1}_{\text{conv},n}(q)$, which typically requires numerical methods to compute.

The basis of the convolution estimators is the weak convergence of the geometric sum S_ϵ in (1) (scaled by its mean) to an exponential as $\epsilon \to 0$, which can hold even when the exponential limit for $T_\epsilon = S_\epsilon + V_\epsilon$ in (2) does not. This can happen when V_ϵ is not "negligible" compared to S_ϵ, as occurs, e.g., for the model described in [23, Section 5]. By explicitly taking into consideration the contribution of V_ϵ to T_ϵ, the convolution estimator can then have smaller bias than its exponential counterpart, as seen in Fig. 5 of [5].

Constructing (biased, as discussed in the last paragraph of Sect. 2.3) CIs based on the convolution estimators requires that the estimators obey corresponding CLTs (with centering constants derived from $\widetilde{G}_{\text{exp}} \star H$ rather than $G \star H$), which we have not yet established (but are working on). This statement even applies for batching CIs, as they also rely on an underlying CLT for each batch. A complication in establishing such CLTs is that in contrast to, e.g., the exponential cdf estimator in (8), the convolution cdf estimator is not simply a function of sample means, so the delta method does not directly apply.

3 Array-RQMC Implementation of Regenerative-Simulation-Based Estimators of Quantiles

3.1 RQMC and Array-RQMC

Quasi-Monte Carlo (QMC) is a deterministic numerical integration method (usually considered over the s-dimensional unit cube $[0, 1]^s$ without much loss of generality) to approximate an integral $I \equiv \int_{[0,1]^s} f(x)\,\mathrm{d}x$ of a given function f. QMC approximates I by an average of evaluations of f over m values from a deterministic sequence $\mathcal{P} = (\theta_i)_{1 \le i \le m}$ of points from $[0, 1]^s$; i.e., the QMC estimator of the integral is $\frac{1}{m}\sum_{i=1}^{m} f(\theta_i)$. The sequence $\mathcal{P}$ of points is designed to "evenly" cover the space $[0, 1]^s$ and is known as a *low-discrepancy sequence*. The most common constructions are lattice points and digital nets, including Sobol' sequences [2, 25]. Error bounds exist [25] under restrictive assumptions, showing that the QMC error shrinks at a rate in $O(m^{-1}(\log m)^s)$ as $m \to \infty$ (and sometimes even faster), better than the $O(m^{-1/2})$ convergence rate of MC's root-mean-square error. (For non-negative functions g_1 and g_2, "$g_1(m) = O(g_2(m))$ as $m \to \infty$" means there are positive constants c and m_0 such that $g_1(m) \le c g_2(m)$ for all $m \ge m_0$.) But applying such bounds is impractical: they are very difficult to compute and can be extremely loose for a given integrand f or a given value of m. RQMC, which has several advantages over QMC, randomizes the sequence $\mathcal{P}$ such that each point of the sequence is uniformly distributed over $[0, 1]^s$ but the points are correlated and keep the low discrepancy to gain the improved convergence rate with respect to MC [13]. We can apply a central

limit theorem on $r \to \infty$ i.i.d. randomizations to obtain a confidence interval for I (see [24] for conditions).

QMC and RQMC efficiency is sensitive to the problem's dimension s (or actually to the *effective dimension* representing the number of coordinates encompassing "most of the variability" of the problem; see [26] for more details). But in a naive implementation of (R)QMC to simulate paths of a Markov chain, the dimension s corresponds to the maximum length of a simulated path, which can be large, even infinite in many cases (as when generating paths up to an unbounded hitting time). In most such situations, RQMC is typically considered useless, yielding no improvement with respect to MC except if the effective dimension is small, which happens only in restricted cases.

To cope with this dimensionality issue, the array-RQMC method has been designed in [16] and further developed in [17] to adapt RQMC to the simulation of Markov chains. As a randomization of the deterministic QMC version presented in [12], array-RQMC simulates a Markov chain $(X_j,\ j \geq 0)$ defined on a state space $\mathcal{S}$ as follows. It assumes a total ordering function h of states in $\mathcal{S}$. Let the initial state X_0 be distributed according to some distribution ν_0. (In our regenerative setting of Sect. 2, we will assume that ν_0 is degenerate, so there is a single fixed starting state, but we recall here the nondegenerate-ν_0 version introduced in [16] for sake of generality.) Transitions of the chain are defined by the stochastic recurrence

$$X_j = \varphi(X_{j-1}, U_j), \quad (j \geq 1), \tag{12}$$

for a given transition kernel φ, where U_j (independent for different j) is a random vector uniformly distributed over $[0, 1)^d$, meaning that d uniforms are used to simulate a single transition step.

While MC typically simulates n chains sequentially and independently, array-RQMC instead generates m chains *in parallel*, simulating the jth step of all the m paths in a negatively correlated way (to reduce the variance in the estimation) before moving to the next step for each path. For $i = 1, 2, \ldots, m$, let $(X_{i,j} : j = 0, 1, 2, \ldots)$ be the ith path generated, with $X_{i,j}$ as the state visited after the jth step. To begin, m initial states $X_{i,0}$ (for $i = 0, \ldots, m$) are generated from the initial distribution ν_0 using an RQMC point set $\mathcal{P}_{m,0} = \{U_{0,0}, \ldots, U_{m-1,0}\}$ in $[0, 1)^{d_0}$ (that is, at most d_0 uniforms are used to generate an initial state); from the property of RQMC points being well distributed over the space, this results in m "well spread" (according to ν_0) initial points for the m chains. The m chains are then sorted (say in increasing order of their state) according to h. Then for the transitions from step $j-1$ to step j (for $j \geq 1$), the next state for each of the m chains is sampled from the previously sorted ones. An RQMC point set $\mathcal{P}_{m,j} = \{U_{0,j}, \ldots, U_{m-1,j}\}$ in $[0, 1)^d$ independent from previous RQMC point sets is used such that for all $i \in \{1, \ldots, m\}$, the transition of the i-th (ordered) chain is generated using the i-th point of $\mathcal{P}_{m,j}$:

$$X_{i,j} = \varphi(X_{i,j-1}, U_{i,j}).$$

And again the states are re-ordered according to h. The process is iterated up to the end of the paths. If the chains have different stopping times, the below algorithm ignores those terminated paths (and not simulated anymore) and their states are specified as ∞ (an absorbing state used to indicate that those simulated paths have already reached the stopping time.)

The algorithm can therefore be described as follows for a discrete-time Markov chain, which we will later modify (at the end of Sect. 3.2) to handle a continuous-time Markov chain, as needed for the array-RQMC convolution estimator:

Array-RQMC algorithm [16]:
1 (Initialization).
Generate a RQMC point set, $\mathcal{P}_0 = \{U_{0,0}, \ldots, U_{m-1,0}\} \subset [0,1)^{d_0}$;
$\forall\, i \in \{0, 1, \ldots, m-1\}$, generate $X_{i,0}$ from $U_{i,0}$;
2 (Simulate chains).
Simulate in parallel m copies of the chain, numbered $0, \ldots, m-1$, as follows:
For $(j = 1; X_{0,j-1} < \infty; j{+}{+})$
Generate an RQMC point set $\mathcal{P}_{m,j} = \{U_{0,j}, \ldots, U_{m-1,j}\} \subset [0,1)^d$
(independent of previous ones);
For all non-terminated chains i, let $X_{i,j} = \varphi_j(X_{i,j-1}, U_{i,j})$;
For terminated chains (i.e., stopping time reached), set $X_{i,j} = \infty$;
Sort (and renumber) the chains for which $X_{i,j} < \infty$ by increasing
order of their states (based on the ordering function h);
(The sorted states $X_{0,j}, \ldots, X_{n-1,j}$ result in an estimator $\hat{F}_j$
of the cdf F_j of the chain at the jth step X_j.)
3 (Output).
Return the estimator obtained from the m generated paths.

The algorithm simulates each transition step across the m chains according to an RQMC point set with good coverage properties over the sampling space. The re-ordering helps to obtain an empirical cdf of the random variable X_j at the $(j-1)$-th step of the chain, so that the RQMC point set at step j is actually generating step-j values from this empirical cdf, from a $(d+1)$-dimensional point set where the first coordinate of the i-th point is i/m and the d other coordinates $U_{i,j}$ (see [16, 17]).

But the main advantage of using array-RQMC with respect to traditional RQMC techniques is that the dimension of the RQMC point sets is $\max(d, d_0)$ for array-RQMC, as compared to $d_0 + d \times \tau'$ for traditional RQMC, where τ' is an upper bound (possibly infinite) for the stopping time τ. Hence, array-RQMC drastically reduces the dimension, from which efficiency improvements can be expected. Actually it is shown in [16, 17] that if stratified sampling is used, the variance of a mean estimator can be $O(m^{-3/2})$ as $m \to \infty$, much faster than the $O(m^{-1})$ for MC. In fact, [17, 20] present numerical results that suggest variances can even shrink as $O(m^{-2})$.

We can easily obtain a confidence interval by considering $r \geq 2$ independent replications (i.e., randomizations) of groups of m chains.

The algorithm is sensitive to the choice of ordering function h. When the state space $\mathcal{S}$ is a (one-dimensional) subset of $\mathbb{R}$, the states have a natural order. But

difficulties arise for state spaces of higher dimension, for which it may not be obvious how to design an effective ordering of the states. This issue is related to that of defining an importance function for the levels in the splitting technique in rare-event simulation [15]: an effective ordering is problem-specific, depending on both the stochastic model and on what is being estimated.

3.2 Array-RQMC Exponential and Convolution Estimators

We explain now how we propose to apply array-RQMC to the exponential and convolution estimators of Sect. 2. Recall that we start in a fixed regenerative state so the initial distribution ν_0 in the array-RQMC is degenerate; no sampling is required to specify the initial state. We start with the exponential estimator in Definition 1 of Sect. 2.3. Recall that this estimator exponentiates the ratio of the estimators $\widehat{\zeta}_n$ and $\widehat{p}_n$ in Eq. (7), with $\widehat{\zeta}_n$ an average over n_{CS} cycles and $\widehat{p}_n$ averaging over n_{IS} cycles, where n_{CS} and n_{IS} may differ. For array-RQMC, we propose to consider a set of (a fixed number) m chains generated in parallel and to apply r_{CS} and r_{IS} independent randomizations of groups of m chains for estimating ζ and p, respectively, from (4). Because (R)QMC methods often work best for point sequences $\mathcal{P}$ of certain specific sizes (e.g., powers of 2), the array-RQMC exponential estimator specifies the same number m of chains for CS and IS, but the randomizations for CS and IS allow for unequal allocations (i.e., different r_{CS} and r_{IS}). By applying independent sets of replications to estimate ζ and p, we are able to estimate the variance of the exponential estimator and construct a (biased; see the discussion at the end of Sect. 2.3) confidence interval based on a CLT (with $1 - e^{-t/\mu}$ as the centering constant due to the bias from fixing $\epsilon > 0$ in (3)), provided $r_{\mathrm{CS}} \to \infty$ and $r_{\mathrm{IS}} \to \infty$.

Formally, denote by $\widehat{\zeta}_m^{(k)}$ ($k \in \{1, \ldots, r_{\mathrm{CS}}\}$) and $\widehat{p}_m^{(k)}$ ($k \in \{1, \ldots, r_{\mathrm{IS}}\}$) as the estimators of ζ and p respectively for the k-th independent group of cycles sampled from array-RQMC. Specifically, we have

$$\widehat{\zeta}_m^{(k)} = \frac{1}{m} \sum_{i=1}^{m} T_i^{(k)} \wedge \tau_i^{(k)}$$

with $T_i^{(k)} \wedge \tau_i^{(k)}$ the minimum of the hitting time and cycle length for the i-th generated array-RQMC chain of the k-th independent replication of groups under crude simulation. Also, we get

$$\widehat{p}_m^{(k)} = \frac{1}{m} \sum_{i=1}^{m} \mathfrak{I}(T_i^{'(k)} < \tau_i^{'(k)}) L_i^{'(k)},$$

with $\mathcal{I}(T_i^{'(k)} < \tau_i^{'(k)})$ and $L_i^{'(k)}$ as the indicator of hitting $\mathcal{A}$ before regenerating and the likelihood ratio, respectively, for the i-th generated array-RQMC chain of the k-th independent replication of groups under IS.

The estimators of ζ, p and mean hitting time μ are then

$$\widehat{\zeta}_{m,r}^{\text{aRQMC}} = \frac{1}{r_{\text{CS}}} \sum_{k=1}^{r_{\text{CS}}} \widehat{\zeta}_m^{(k)}, \quad \widehat{p}_{m,r}^{\text{aRQMC}} = \frac{1}{r_{\text{IS}}} \sum_{k=1}^{r_{\text{IS}}} \widehat{p}_m^{(k)}, \quad \widehat{\mu}_{m,r}^{\text{aRQMC}} = \frac{\widehat{\zeta}_{m,r}^{\text{aRQMC}}}{\widehat{p}_{m,r}^{\text{aRQMC}}},$$

from which the *array-RQMC exponential estimator* of the cdf $F(t)$ of T is

$$\widehat{F}_{\text{exp},m,r}^{\text{aRQMC}}(t) = 1 - e^{-t/\widehat{\mu}_{m,r}^{\text{aRQMC}}}. \tag{13}$$

From the independent replications of groups of m parallel chains, we can obtain variance estimators of $\widehat{\zeta}_{m,r}^{\text{aRQMC}}$, $\widehat{p}_{m,r}^{\text{aRQMC}}$, and $\widehat{\mu}_{m,r}^{\text{aRQMC}}$, leading to a (biased) CI for $F(t)$ derived similarly to what is done for MC in [4] from the CLT described in the last paragraph of Sect. 2.3, where an estimator of the asymptotic variance ψ_t^2 can be computed from the sample variances of $\widehat{\zeta}_m^{(k)}$, $k = 1, 2, \ldots, r_{\text{CS}}$, and $\widehat{p}_m^{(k)}$, $k = 1, 2, \ldots, r_{\text{IS}}$.

We specify an allocation of the $r = r_{\text{CS}} + r_{\text{IS}}$ independent groups of chains between the crude and IS simulations with $r_{\text{CS}} = \gamma' r$ and $r_{\text{IS}} = (1 - \gamma')r$ for a user-specified constant $\gamma' \in (0, 1)$. From a pre-simulation, we can choose γ' with the goal to minimize the work-normalized variance [14] of the mean-hitting-time estimator $\widehat{\mu}_{m,r}^{\text{aRQMC}}$, similarly to what is done for MC [4, 6]. The optimal allocation parameter γ' for array-RQMC can differ from γ for MC in Sect. 2.3.

As explained in the last paragraph of Sect. 2.4, providing a CI (even with batching) using the convolution estimator requires a CLT, which we have not yet established for MC (although we are currently working on it). If we then decide to forgo a CI based on the convolution estimator, then we could just consider a single group (i.e., $r = 1$) to decompose the full budget $n = r \times m = m$ into m_{CS} and m_{IS} with $m_{\text{CS}} + m_{\text{IS}} = m$. But then the convolution estimator has an unfair advantage over the exponential estimator with the same total budget because the former is based on a larger QMC point sequence (and QMC has faster convergence than MC, which corresponds to the randomizations). As such, our numerical experiments in Sect. 4 construct the convolution estimator with the same allocation (with r_{CS} and r_{IS}) that is used for the exponential estimator.

As studied in [17, 20], the efficacy of array-RQMC depends critically on the choice of the ordering function h, but we do not pursue that issue here. When the state space $\mathcal{S}$ is a one-dimensional subset of $\mathbb{R}$, as in the M/M/1 example that we will study numerically in Sect. 4, there is a natural ordering of states, which can be effective.

Recall also that d is the number of uniforms employed to simulate a single transition step in (12). The exponential estimator in Definition 1 requires estimating only the mean μ in (4), so discrete-time conversion [3, 9] can be applied. Specifically, to estimate μ, we need to generate only the embedded discrete-time Markov chain

(DTMC), replacing the exponential holding times in each successive state visited by its conditional mean (given the DTMC). In addition to reducing the number of uniforms needed to generate each transition, discrete-time conversion (as a form of conditional Monte Carlo) also reduces (asymptotic) variance. Thus, generating a path of the DTMC typically has $d = 1$ in (12), as a single DTMC step requires only a single uniform, even if for some applications using $d > 1$ may lead to more efficient implementations. But for the convolution estimator, discrete-time conversion cannot be applied when estimating the cdf H in (11) since it further requires the actual exponential holding times in each state visited. Thus, the number of uniforms needed to generate each transition is in this case $d' = d + d_g$, where d_g stands for the number of uniforms to generate the random holding time once the new state is selected. In our examples, we will typically have $d_g = 1$, those times being exponentially distributed, generated from the inversion procedure of a single uniform.

4 Numerical Illustration of the Gain on the Simulation of an M/M/1 Queue

To study the effectiveness of array-RQMC, consider the simulation of an M/M/1 queue, also studied in [16], with arrival rate $\lambda = 1.0$ and service rate $\mu' = 4.0$. For the process $(X(t) : t \in \mathbb{R}_+)$ with $X(t)$ denoting the total number of customers in the system at time t and $X(0) = 0$, our goal is to estimate quantiles and the cdf F of the hitting time T to a given buffer size N. As in [4], we apply MSIS, where the IS swaps the arrival and service rates, an approach known to be efficient when using the ratio estimator (7) of the mean hitting time as the buffer size increases, and therefore hitting times typically increase too. As explained in Sect. 3.2, the exponential and convolution estimators will use $r = r_{\text{CS}} + r_{\text{IS}}$ independent sets of randomizations of m parallel chains to estimate the variances of $\widehat{\zeta}_{m,r}^{\text{aRQMC}}$ and $\widehat{p}_{m,r}^{\text{aRQMC}}$ and obtain a (biased; see the last paragraph of Sect. 2.3) confidence interval. All the results are each time compared with the MC exponential and convolution estimators with a total of $n = m \times r$ MSIS cycles.

Our experiments test different sets of RQMC point sets, among classical ones:

- Sobol' with a left matrix scrambling (named Matoušek scrambling [22]);
- Randomly-shifted lattice rule [18, 31] with lattice points selected using [19];
- (The same) Randomly-shifted lattice rule plus baker's transformation [8];
- Randomly-shifted Sobol' sequence (often yielding good numerical results, see for example [30]).

Table 1 displays the outputs for three different quantiles $F^{-1}(q)$ (when $q = 0.1$, 0.5 and 0.9). For constructing the exponential estimator, array-RQMC uses $r = 100$ independent randomizations of $m = 2^{14}$ parallel chains, compared with $r \times m$ cycles for MC. To simplify the following discussion about array-RQMC, we will focus on the exponential estimator (but similar comments also apply to the convolution

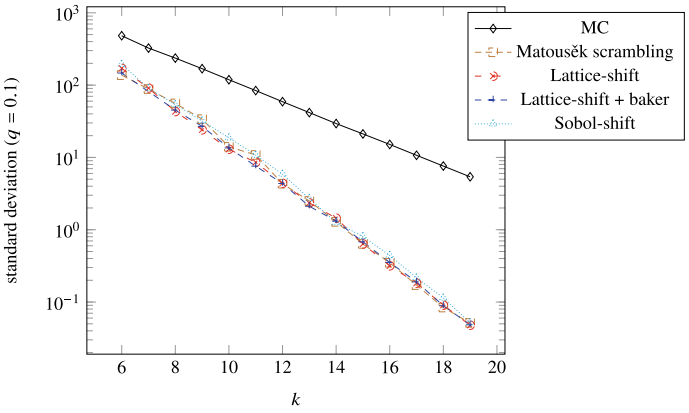


Fig. 2 Standard deviation of array-RQMC exponential estimators as a function of $m = 2^k$, for the M/M/1 queue with $\lambda = 1.0$ and $\mu' = 4.0$ when estimating q-quantiles of hitting times to $N = 10$ with $r = 128$. For MC, we consider $n = m \times r$

estimator). The array-RQMC exponential q-quantile estimator applies array-RQMC to independently estimate ζ by CS and p by IS to handle the ratio μ from (4). The performance of an array-RQMC estimator depends critically on the choice of the ordering function h (Sect. 3), which should be tailored for the particular estimand. We could try to select different h for ζ and p (see [15, 17] for discussions on this), taking into account, e.g., the accumulated "reward" (time already spent for CS or accumulated likelihoood ratio for IS) to which an approximation of the remaining reward is appended. But our experiments instead simply had that CS and IS used the same ordering function h (the number of customers in the system), which gave similar results.

Column 5 of Table 1 shows that compared to MC for the same total number of cycles generated, array-RQMC drastically reduces the variance of the estimators for each quantile level q. The variance-reduction factor (i.e., ratio of variances for MC and array-RQMC) is always well over 100, with the specific amount depending on the randomization technique and choice of the low-discrepancy sequence. For this example, the array-RQMC variances differ by up a factor of 4, with Matoušek scrambling and randomly shifted Sobol' sequence the most effective. From the numerically computed exact quantile values 4.91036e+04 for $q = 0.1$, 3.230287e+05 for $q = 0.5$ and 1.073074e+06 for $q = 0.9$, we see that array-RQMC estimators are more accurate than MC ones, and all competitive. Convolution estimators are accurate as well, expected to reduce the existing bias with respect to exponential ones [4, 5].

Figure 2 displays in a log-log scale the standard deviation of the exponential estimators in terms of $m = 2^k$ with fixed $r = 128$ for the various array-RQMC methods as well as for MC with $n = r \times m$ total cycles. We display only the results for the $q = 0.1$ quantile since all other quantiles have the same curve up to a multiplicative constant.

Table 1 Results for the M/M/1 queue with $\lambda = 1.0$, $\mu' = 4.0$ when estimating q-quantiles $F^{-1}(q)$ of hitting times to $N = 10$. We consider $r = 100$ and $m = 2^{14}$ for RQMC and a total of $r \times m$ cycles for MC. Exact values are 4.91036e+04 for $q = 0.1$, 3.230287e + 05 for $q = 0.5$ and 1.073074e+06 for $q = 0.9$

q	Method	Exp. Est.	Conf. Interval	Variance	Conv. Est.
0.1	MC	4.9056e+04	(4.8990e+04, 4.9121e+04)	1.12e+03	4.9146e+04
0.1	Matoušek scrambling	4.9102e+04	(4.9099e+04, 4.9104e+04)	2.08e+00	4.9102e+04
0.1	Lattice-shift	4.9104e+04	(4.9099e+04, 4.9110e+04)	7.81e+00	4.9105e+04
0.1	Lattice-shift + baker	4.9104e+04	(4.9099e+04, 4.9109e+04)	5.66e+00	4.9100e+04
0.1	Sobol-shift	4.9102e+04	(4.9099e+04, 4.9105e+04)	2.14e+00	4.9101e+04
0.5	MC	3.2273e+05	(3.2230e+05, 3.2316e+05)	4.85e+04	3.2330e+05
0.5	Matouseˇk scrambling	3.2303e+05	(3.2301e+05, 3.2305e+05)	8.98e+01	3.2301e+05
0.5	Lattice-shift	3.2305e+05	(3.2301e+05, 3.2309e+05)	3.38e+02	3.2303e+05
0.5	Lattice-shift + baker	3.2305e+05	(3.2302e+05, 3.2308e+05)	2.45e+02	3.2300e+05
0.5	Sobol-shift	3.2303e+05	(3.2301e+05, 3.2305e+05)	9.25e+01	3.2300e+05
0.9	MC	1.0721e+06	(1.0706e+06, 1.0735e+06)	5.36e+05	1.0740e+06
0.9	Matouseˇk scrambling	1.0731e+06	(1.0730e+06, 1.0731e+06)	9.91e+02	1.0730e+06
0.9	Lattice-shift	1.0731e+06	(1.0730e+06, 1.0733e+06)	3.73e+03	1.0731e+06
0.9	Lattice-shift + baker	1.0731e+06	(1.0730e+06, 1.0732e+06)	2.70e+03	1.0730e+06
0.9	Sobol-shift	1.0731e+06	(1.0730e+06, 1.0732e+06)	1.02e+03	1.0730e+06

All array-RQMC estimators are of the same order of magnitude and outperform the MC one. Larger m yields greater variance reduction with respect to MC, as expected due to the benefit of the generated sequences' low discrepancy.

The log-log curves in Fig. 2 are close to linear. It is interesting to investigate the convergence rate of the standard deviation in terms of m. A standard procedure for convergence-rate estimation of QMC and RQMC methods applies log-log regression. Assume that the standard deviation σ_m as a function of m satisfies $\sigma_m \approx am^{-b}$ for some $a, b > 0$, which is equivalent to

Table 2 Log-log regression corresponding to values of Fig. 2 for the standard deviation of array-RQMC and MC exponential estimators as a function of m, on the M/M/1 queue model with $\lambda = 1.0$, $\mu' = 1.0$ when estimating q-quantiles for of hitting times to $N = 10$. For MC, we consider $n = r \times m$

Method	$q = 0.1$
MC	$3702 \times m^{-0.4965}$
Matoušek scrambling	$7943 \times m^{-0.9055}$
Lattice-shift	$7068 \times m^{-0.8971}$
Lattice-shift + baker	$6615 \times m^{-0.8902}$
Sobol-shift	$8470 \times m^{-0.8969}$

$$\ln(\sigma_m) \approx \ln(a) - b \ln(m).$$

Applying a classical regression for the values of $m = 2^k$ with $k \in \{6, 7, \ldots, 19\}$ in Fig. 2 leads to the regression coefficients in Table 2.

The table verifies the $m^{-0.5}$ convergence of MC. For all array-RQMC techniques, the standard deviation shrinks at about rate $m^{-0.9}$, much faster than MC.

From the known exact values for this M/M/1 model, Fig. 3 displays in a log-log scale the error of the exponential *and convolution* estimators in terms of m with $r = 128$ fixed for the various array-RQMC methods as well as for MC with $n = r \times m$ ($r \times m$ is also used for all convolution estimators). Each plotted point is the average of the absolute errors obtained over $K = 10$ independent replications to smooth the curves with respect to drawing the error for a single replication.

Figure 3 shows that all array-RQMC techniques are of the same order of magnitude of accuracy, and order(s) of magnitude better than the corresponding MC accuracy. Also, for the "extreme" quantiles with $q = 0.1$ and $q = 0.9$, as $m = 2^k$ increases, the array-RQMC errors for the estimators seem to stabilize and converge to a positive (even if small) value, which results from the rarity parameter $\epsilon > 0$ being fixed in (3) because N is fixed. This suggests that the standard deviation is becoming negligible with respect to bias for the exponential estimation, meaning that the bias for the exponential estimator is larger than for the convolution estimator; this is more visible when q is small, e.g., for $q = 0.1$ (see also [5, Fig. 5]). (Also see the related discussion for MC in the last two paragraphs of Sect. 2.3.) The stabilizing constant seems smaller for the convolution estimators than for the exponential ones, indicating a smaller bias for the convolution estimator. As noted before for MC in the penultimate paragraph of Sect. 2.4, the convolution estimator more explicitly accounts for the contribution of V to $R = S + V$ in (1) than the exponential estimator.

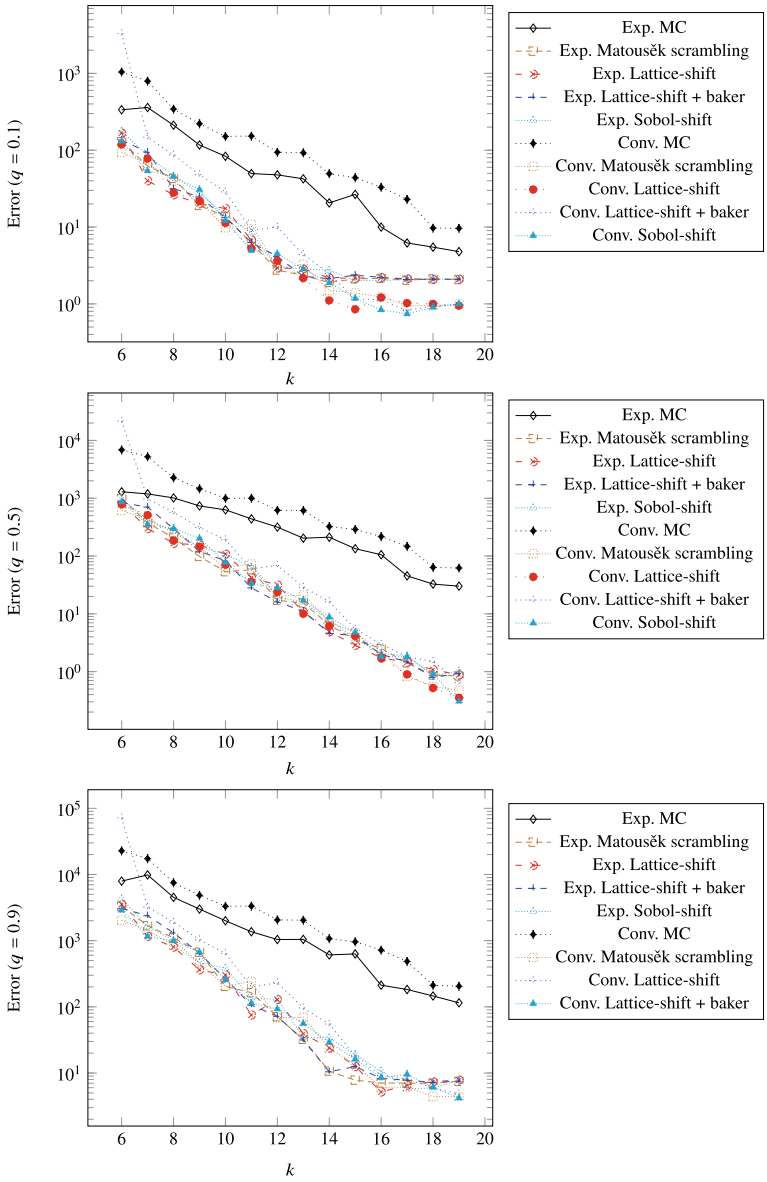


Fig. 3 Errors of the various exponential and convolution estimators as a function of m, for the M/M/1 queue with $\lambda = 1.0$, $\mu' = 4.0$ when estimating q-quantiles for of hitting times to $N = 10$ with $r = 128$. For MC, we consider $n = m \times r$. Each plotted point is the average of $K = 10$ independent replications

5 Conclusions

Estimating distributions of hitting times by simulation presents substantial challenges, especially when related to rarely visited sets. We previously designed [4, 5] MC methods to estimate distributions and quantiles of hitting times in a regenerative context, when hitting the rare set before regeneration is rare. Based on the limiting behavior of a geometric distribution converging to an exponential as when the success probability tends to zero, [4, 5] design two "simple" estimators using previous importance sampling designed to compute means. We proposed in this paper to combine the estimators with array-RQMC, a simulation method simulating paths of the Markov chain in parallel and distributing the sample points to cover more efficiently the space, hence reducing variance. We have illustrated on a standard example that the combination can reduce the variance by several orders of magnitude.

There are nevertheless several questions warranting further study. As noted in the last two paragraphs of Sect. 2.3, variance is not the only component of the simulation error. There is also bias coming from the exponential approximations, which become exact as the rarity parameter $\epsilon \to 0$ in, e.g., (2), but in practice we always have a fixed $\epsilon > 0$, resulting in bias in (3). Since array-RQMC can substantially reduce the variance, bias may significantly contribute to the estimator's mean-squared error, and increasing the sample size will not eliminate this source of bias. Thus, increasing the number m of parallel chains in array-RQMC can provide benefits up to a point, but eventually, bias from fixed $\epsilon > 0$ becomes the dominant issue. This issue deserves further study. Also, array-RQMC efficiency depends on the dimension of the state space $\mathcal{S}$ and of the RQMC point set. More investigations on this are required.

References

1. Ben Abdellah, A., L'Ecuyer, P., Owen, A., Puchhammer, F.: Density estimation by randomized quasi-Monte Carlo. SIAM J. Uncertain. Quantif. **9**(1), 280–301 (2021)
2. Dick, J., Pillichshammer, F.: Digital Nets and Sequences: Discrepancy Theory and Quasi-Monte Carlo Integration. Cambridge University Press, Cambridge, U.K. (2010)
3. Fox, B.L., Glynn, P.W.: Discrete time conversion for simulating semi-Markov processes. Oper. Res. Lett. **5**, 191–196 (1986)
4. Glynn, P.W., Nakayama, M.K., Tuffin, B.: Using simulation to calibrate exponential approximations to tail-distribution measures of hitting times to rarely visited sets. In: Proceedings of the 2018 Winter Simulation Conference. Institute of Electrical and Electronics Engineers, Piscataway, NJ (2018)
5. Glynn, P.W., Nakayama, M.K., Tuffin, B.: Comparing regenerative-simulation-based estimators of the distribution of the hitting time to a rarely visited set. In: Bae, K.H., Feng, B., Kim, S., Lazarova-Molnar, S., Zheng, Z., Roeder, T., Thiesing, R. (eds.) Proceedings of the 2020 Winter Simulation Conference. IEEE, Piscataway, New Jersey (2020)
6. Goyal, A., Shahabuddin, P., Heidelberger, P., Nicola, V., Glynn, P.W.: A unified framework for simulating Markovian models of highly dependable systems. IEEE Trans. Comput. **C-41**(1), 36–51 (1992)
7. Heidelberger, P.: Fast simulation of rare events in queueing and reliability models. ACM Trans. Model. Comput. Simul. **5**, 43–85 (1995)

8. Hickernell, F.J.: Obtaining $O(N^{-2+\epsilon})$ convergence for lattice quadrature rules. In: Fang, K.T., Hickernell, F.J., Niederreiter, H. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2000, pp. 274–289. Springer, Berlin (2002)
9. Hordijk, A., Iglehart, D.L., Schassberger, R.: Discrete-time methods for simulating continuous-time Markov chains. Adv. Appl. Probab. **8**, 772–788 (1976)
10. Kalashnikov, V.: Topics on Regenerative Processes. CRC Press, Boca Raton (1994)
11. Kalashnikov, V.: Geometric Sums: Bounds for Rare Events with Applications. Kluwer Academic Publishers, Dordrecht, The Netherlands (1997)
12. Lécot, C., Tuffin, B.: Quasi-Monte Carlo methods for estimating transient measures of discrete time Markov chains. In: Niederreiter, H. (ed.) Monte Carlo and Quasi-Monte Carlo Methods 2002, pp. 329–343. Springer, Berlin (2004)
13. L'Ecuyer, P.: Randomized quasi-Monte Carlo: an introduction for practitioners. In: Glynn, P.W., Owen, A.B. (eds.) Monte Carlo and Quasi-Monte Carlo Methods: MCQMC 2016, pp. 29–52. Springer, Berlin (2018)
14. L'Ecuyer, P., Blanchet, J.H., Tuffin, B., Glynn, P.W.: Asymptotic robustness of estimators in rare-event simulation. ACM Trans. Model. Comput. Simul. **20**(1), Article 6 (2010)
15. L'Ecuyer, P., Demers, V., Tuffin, B.: Rare-events, splitting, and quasi-Monte Carlo. ACM Trans. Model. Comput. Simul. **17**(2), Article 9, 45 (2007)
16. L'Ecuyer, P., Lécot, C., Tuffin, B.: Randomized quasi-Monte Carlo simulation of Markov chains with an ordered state space. In: Niederreiter, H., Talay, D. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2004, pp. 331–342. Springer, Berlin (2006)
17. L'Ecuyer, P., Lécot, C., Tuffin, B.: A randomized quasi-Monte Carlo simulation method for Markov chains. Oper. Res. **56**(4), 958–975 (2008)
18. L'Ecuyer, P., Lemieux, C.: Variance reduction via lattice rules. Manage. Sci. **46**(9), 1214–1235 (2000)
19. L'Ecuyer, P., Marion, P., Godin, M., Fuchhammer, F.: A tool for custom construction of QMC and RQMC point sets. In: Monte Carlo and Quasi-Monte Carlo Methods: MCQMC 2020 (2020). https://arxiv.org/abs/2012.10263
20. L'Ecuyer, P., Munger, D., Lécot, C., Tuffin, B.: Sorting methods and convergence rates for Array-RQMC: some empirical comparisons. Math. Comput. Simul. **143**, 191–201 (2018)
21. L'Ecuyer, P., Puchhammer, F., Ben Abdellah, A.: Monte Carlo and quasi-Monte Carlo density estimation via conditioning. INFORMS J. Comput. (2021). To appear. arXiv:1906.04607
22. Matoušek, J.: On the L_2-discrepancy for anchored boxes. J. Complex. **14**, 527–556 (1998)
23. Nakayama, M.K., Tuffin, B.: Efficient estimation of the mean hitting time to a set of a regenerative system. In: Mustafee, N., Bae, K.H., Lazarova-Molnar, S., Rabe, M., Szabo, C., Haas, P., Son, Y.J. (eds.) Proceedings of the 2019 Winter Simulation Conference, pp. 416–427. Institute of Electrical and Electronics Engineers, Piscataway, New Jersey (2019)
24. Nakayama, M.K., Tuffin, B.: Sufficient conditions for central limit theorems and confidence intervals for randomized quasi-Monte Carlo methods. Techreport hal-03196085, INRIA (2021). https://hal.inria.fr/hal-03196085
25. Niederreiter, H.: Random Number Generation and Quasi-Monte Carlo Methods, vol. 63. SIAM, Philadelphia (1992)
26. Owen, A.B.: Latin supercube sampling for very high-dimensional simulations. ACM Trans. Model. Comput. Simul. **8**(1), 71–102 (1998)
27. Rubino, G., Tuffin, B. (eds.): Rare Event Simulation using Monte Carlo Methods. Wiley, Chichester, UK (2009)
28. Sadowsky, J.S.: Large deviations theory and efficient simulation of excessive backlogs in a GI/GI/*m* queue. IEEE Trans. Autom. Control **36**, 1383–1394 (1991)
29. Shahabuddin, P.: Importance sampling for highly reliable Markovian systems. Manage. Sci. **40**(3), 333–352 (1994)
30. Tuffin, B.: On the use of low discrepancy sequences in Monte Carlo methods. Monte Carlo Methods Appl. **2**(4), 295–320 (1996)
31. Tuffin, B.: Variance reduction order using good lattice points in Monte Carlo methods. Computing **61**(4), 371–378 (1998)

Foundations of Ranking & Selection for Simulation Optimization

Barry L. Nelson

Abstract In addition to his voluminous and profound research accomplishments, Pierre L'Ecuyer is an extraordinary educator; this includes expository talks and papers, especially in the area of pseudorandom-number generation. This paper is written in that same spirit, covering the foundations of ranking & selection for simulation optimization; simulation optimization is also an area of exceptional accomplishment for Pierre.

Keywords Ranking & selection · Stochastic optimization · Bayesian optimization · Multi-armed bandits · Parallel simulation

1 Introduction

Suppose that we have the ability to simulate $k = 4$ different system designs that use redundancy to be resistant to system failure. Let $Y(x)$ be the time to failure of design type $x = 1, 2, 3, 4$. Your job, as the analyst, is to use the simulation to find $x^\star = \text{argmax}_x \text{E}[Y(x)]$, the system design leading to the largest mean time to failure. How would you do this?

The field of ranking & selection (R&S) provides procedures that "solve" problems of this type. Features we might like in a R&S procedure include controlling the number of simulation replications automatically; providing statistical guarantees of correctness; being appropriate for large as well as small numbers of systems, k; the facility to exploit modern parallel computing; and to do all of this computationally and statistically efficiently.

The field of *simulation optimization* (SO)—of which R&S is a part—attacks stochastic optimization problems in which the objective function is some property of the output of a stochastic, often dynamic and non-stationary, simulation. Critically, the property of interest can only be estimated by simulating instances (feasible solutions, system designs), and those simulations may be computationally expensive. All

B. L. Nelson (✉)
Northwestern University, Evanston, IL, USA
e-mail: nelsonb@northwestern.edu

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_18

SO algorithms are subject to three sources of error: They may fail to simulate the optimal solution; they may fail to recognize the best solution that was simulated; and they may report an optimistic (biased) estimate of the performance of the solution that they do select. R&S is the only class of SO algorithms that controls all three sources of error, but at the cost of simulating *all* system designs: R&S procedures are exhaustive SO algorithms designed specifically to control statistical error.

R&S originated with Robert Bechhofer (Cornell) and Shanti Gupta (Purdue) in the 1950s to address biostatistics problems such as finding the most efficacious of three drug treatments and a placebo. See [1, 13]. The problem characteristics assumed by early R&S procedures include a small number of treatments, k; normally distributed responses; relatively equal (maybe even known) variances; and a requirement to be easy to implement, for instance by applying treatments to batches of subjects rather than sequentially (e.g., one subject at a time and waiting for the results before deciding the next treatment to apply).

At the 1983 Winter Simulation Conference David Goldsman (Georgia Tech) presented a tutorial on R&S [12], and organized a session with Bechhofer and Gupta, arguing that R&S was useful for optimizing simulated systems. The simulation community quickly embraced this paradigm, but had more expansive objectives than the founders, including much larger numbers of "treatments" (simulated system designs) k; non-normal (nominal) output data; significantly unequal variances across systems; and intentionally induced dependence across systems due to the use of common random numbers. In addition, since data are generated by computer simulations that are easily controlled, simulation researchers and practitioners were not concerned with how complex or sequential the R&S procedure is as long as it is effective (selects the best system design) and computationally efficient (generates as few simulation replications as possible, since the simulation was assumed to be more computationally expensive than the overhead of the R&S procedure).

R&S has been a theoretical and practical success for simulation: There is supporting theory, including asymptotic regimes for non-normal data and effective use of "statistical learning." Further, R&S has been routinely applied to real problems, partly because R&S procedures are included in commercial simulation software. Of course there is a R&S problem-size limit, since all system designs must be simulated. Therefore, much of the research effort in R&S for simulation has been dedicated to extending this limit via enhanced statistical efficiency to reduce simulation effort and parallel computing to speed up execution. See [2, 21] for earlier surveys.

This paper is a significant extension of [27], and a companion to the online masterclass "Ranking & Selection for Simulation Optimization" at http://users.iems.northwestern.edu/~nelsonb/RSMasterclass.html. The web site contains `R` code for all of the R&S procedures described here along with slides, videos and self-paced exercises supporting this tutorial. The purpose of the masterclass and this paper is to present foundations and broad themes in R&S for SO, rather than details or new results. In Sect. 2 we set up the R&S problem. Section 3 describes the "normal means" case, the most widely studied and solved R&S problem. Exploiting parallel

computing in R&S is discussed in Sect. 4. Some formulations beyond normal means are presented in Sect. 5. Finally, in Sect. 6 we briefly contrast R&S with the related field of multi-armed bandits.

2 Set Up

For much of the paper the following set up applies. The true system performance parameters (which are unknown) are $\mu(1) \leq \mu(2) \leq \cdots \leq \mu(k-1) \leq \mu(k)$, and we refer to system k, or any system tied with system k, as "the best." For system x we can estimate $\mu(x)$ with a consistent estimator; for instance, when $\mu(x)$ is the expected value we may employ the sample mean of $n(x)$ replications:

$$\bar{Y}(x) = \frac{1}{n(x)} \sum_{j=1}^{n(x)} Y_j(x)$$

where $Y_j(x)$ is the jth independent and identically distributed (i.i.d.) replication from system design x. We will focus on selecting the best mean, but consider other performance measures in Sects. 5–6. The R&S procedure ultimately returns something like $\widehat{x}^\star = \text{argmax}_{x \in \{1,2,\ldots,k\}} \bar{Y}(x)$ as the selected system. We consider two categories of objectives for the R&S procedure:

1. Fixed Precision: Simulate until a prespecified level of inference is achieved, ideally a probability of correct selection (PCS), defined as $\Pr\{\widehat{x}^\star = k\} \geq 1 - \alpha$. Since this can be computationally impossible, for instance if there are ties for the best, a compromise such as one of the following is accepted, where $\delta > 0$ is a user-specified parameter:

- **Indifference zone**: $\text{PCS} = \Pr\{\widehat{x}^\star = k \mid \mu(k) - \mu(k-1) \geq \delta\} \geq 1 - \alpha$, where "$\mid \mu(k) - \mu(k-1) \geq \delta$" indicates that the guarantee is only for problems in which the means satisfy this inequality. That is, the best system is highly likely to be selected when there is at least a minimum separation between the best and second-best system.
- **Good selection**: $\text{PGS} = \Pr\{\mu(k) - \mu(\widehat{x}^\star) \leq \delta\} \geq 1 - \alpha$. That is, a system with no more than a specified optimality gap is highly likely to be selected.
- **Top** m: $\Pr\{\widehat{x}^\star \in [k, k-1, \ldots, k-m+1]\} \geq 1 - \alpha$. That is, one of the m best systems is highly likely to be selected.
- **Subset**: Find $\widehat{\mathcal{S}} \subseteq \{1, 2, \ldots, k\}$ such that $\Pr\{k \in \widehat{\mathcal{S}}\} \geq 1 - \alpha$. That is, a subset (ideally small) is returned that is highly likely to contain the best system.

These are typically *frequentist* guarantees to be achieved as efficiently as possible.

2. Fixed Budget: Obtain as strong an inference as possible within a given computation budget, often formulated as minimizing some expected loss for the chosen system design, $\text{E}[\mathcal{L}(\widehat{x}^\star, k)]$:

- **0-1 Loss**: Minimize the posterior probability of incorrect selection, $\Pr\{\widehat{x}^\star \neq k|\mathcal{H}\}$.
- **Opportunity cost**: Minimize the posterior expected optimality gap, $\mathrm{E}[\mu(k) - \mu(\widehat{x}^\star)|\mathcal{H}]$.

The inference is typically *Bayesian* in nature, and $\mathcal{H}$ includes the entire history of simulation runs performed and outputs obtained until the budget is exhausted. We will consider both fixed-precision and fixed-budget perspectives in this chapter.

3 The Normal Means Case

The most widely studied case assumes that from system x we can obtain $Y_1(x), Y_2(x), \ldots$ that are i.i.d. normally distributed with mean $\mu(x)$ and variance $\sigma^2(x)$, denoted $\mathrm{N}(\mu(x), \sigma^2(x))$. Further it may be possible to induce $\mathrm{Cov}(Y(x), Y(x')) \neq 0$ if we use common random numbers. Since so much research effort has been expended on this problem, it is reasonable to ask, is normally distributed output actually relevant for simulation problems? Fortunately, the answer is frequently "yes." Each output Y is often the average of *many* more basic outputs, e.g., daily average customer waiting time is the average of many individual customers' waiting times. Also, the sample sizes prescribed by R&S procedures are often large, so we can group or "batch" outputs to obtain approximate normality. And many normal-means procedures are asymptotically valid for non-normal data, as discussed in Sect. 3.8.

Initially we will assume that we can only simulate one system at a time, and then later we parallelize simulations. One-system-at-a-time procedures are often observation-efficient, but may not be computationally efficient in parallel.

3.1 The Indifference-Zone (IZ) Formulation

One of the most well-known IZ procedures is due to [35]:

Rinott's Procedure

1. Choose confidence level $1 - \alpha$, initial sample size $n_0 \geq 2$ and indifference zone parameter $\delta > 0$. Set $h = h(k, 1 - \alpha, n_0)$, a constant that depends on the number of systems, desired confidence level and the initial sample size.
2. For each system $x = 1, 2, \ldots, k$ do the following:
   a. Simulate n_0 replications and compute the sample variance $S^2(x)$.
   b. Compute $N(x) = \left\lceil \dfrac{h^2 S^2(x)}{\delta^2} \right\rceil$
   c. Simulate $\max\{0, N(x) - n_0\}$ additional replications from system x.
   d. Compute the sample mean of all $N(x)$ replications, $\bar{Y}(x)$.
3. Choose $\widehat{x}^\star = \mathrm{argmax}_x \bar{Y}(x)$.

Rinott's procedure assumes that the outputs are i.i.d. normally distributed, have unknown and possibly unequal variances, and are independent across systems. The last assumption implies using distinct random number seeds for each system's simulation. Rinott guarantees

$$\text{PCS} = \Pr\{\widehat{x}^\star = k |\ \mu(k) - \mu(k-1) \geq \delta\} \geq 1 - \alpha.$$

Below we will outline how Rinott-like procedures provide this guarantee. The parameter δ is often interpreted as the "smallest practically significant difference."

Rinott is easy to implement, and because it requires no coordination among systems it is easy to parallelize. However, it is pessimistic: it assumes the means are in the "slippage configuration" $\mu(1) = \mu(2) = \cdots = \mu(k-1) = \mu(k) - \delta$. This pessimisim leads to more simulation than necessary to achieve the desired PCS for many problems in which the means are more favorably spaced. What happens if there are other good (closer than δ) systems? It turns out that Rinott also has a $1 - \alpha$ good selection guarantee, which means selecting a system within δ of the best; this happy fact was not known until more recently [28].

Notice that the sample size $N(x)$ grows as h^2/δ^2. How does $h(k, 1-\alpha, n_0)$ grow with the number of systems k? Answer: too fast to be practical for really large k, so other strategies (described later in this section) are needed for that case.

Rinott-like procedures achieve their guarantee based on some version of the following argument. Since we assume $\mu(k) - \mu(x) \geq \delta$, $x \neq k$, we have

$$\begin{aligned}
&\Pr\left\{\bar{Y}(k) > \bar{Y}(x)\right\} \\
&\qquad = \Pr\left\{\bar{Y}(k) - \bar{Y}(x) > 0\right\} \\
&\qquad = \Pr\left\{\bar{Y}(k) - \bar{Y}(x) - [\mu(k) - \mu(x)] > -[\mu(k) - \mu(x)]\right\} \\
&\qquad \geq \Pr\left\{\bar{Y}(k) - \bar{Y}(x) - [\mu(k) - \mu(x)] > -\delta\right\}.
\end{aligned}$$

The statistic $\bar{Y}(k) - \bar{Y}(x) - [\mu(k) - \mu(x)]$ has mean 0, so we can find the number of replications needed to provide the desired probability guarantee considering only δ and the variances.

This formulation—where we want PCS $\geq 1 - \alpha$ when $\mu(k) - \mu(x) \geq \delta$ and we assume the slippage configuration—has been dominant in frequentist R&S because it frees the probability statement from dependence on the true means. There are two challenges: When $\mu(k) - \mu(x) \gg \delta$ the slippage assumption does not exploit it to gain efficiency, which is particularly critical when k is large. And when $\mu(k) - \mu(x) < \delta$ for some inferior system x, we would like a "good selection" guarantee, which Rinott provides, but this is not the case for all IZ procedures; see Sect. 3.6.

3.2 R&S Based on "Statistical Learning"

The following ideas for R&S are based (formally or informally) on Bayesian reasoning. See [10] for a more complete tutorial.

Frequentist reasoning goes like this: $\mu(1), \mu(2), \ldots, \mu(k)$ are *fixed* performance measures and probability statements (e.g., PCS, PGS) are with respect to repeated independent experiments on the same problem. Bayesian reasoning starts from the premise that we have uncertainty about the problem itself (e.g., which system is the best) that we characterize via a prior probability distribution, and we then reduce our uncertainty by running simulation experiments and updating our prior distribution to a (more informative) posterior (after experiment) distribution using Bayes' rule. Typically the experiment-then-posterior-updating cycle is done repeatedly for many iterations.

In R&S our prior on the true means, and perhaps additional aspects, of the problem is

$$\underbrace{\mu(1), \ldots, \mu(k)}_{\text{your problem}} \quad \sim \quad \underbrace{M(1), \ldots, M(k).}_{\text{r.v.'s with a "prior" distribution}}$$

After observing $(x, Y_j(x))$, we update our knowledge based on the conditional ("posterior") distribution of $[M(1), \ldots, M(k)]$ given the entire history, denoted by $\mathcal{H}$. A generic, fixed-budget, Bayesian R&S procedure is given below. In this procedure $x^{(j)}$ denotes the system we choose to simulate on iteration j of the procedure.

Generic Bayesian R&S

1. For $x \in \{1, 2, \ldots, k\}$, set $n(x) = 0$, $\bar{Y}(x) = \text{null}$, $\mathcal{H}_0 = \emptyset$, $j = 0$.
2. $x^{(j)} = \pi(\mathcal{H}_j)$ and simulate $Y_{j+1}(x^{(j)})$ [policy $\pi(\cdot)$ based on the posterior distribution].
3. Update $n(x^{(j)}) = n(x^{(j)}) + 1$ and $\bar{Y}(x^{(j)}) = \dfrac{1}{n(x^{(j)})} \displaystyle\sum_{i:\, x^{(i)} = x^{(j)}} Y_{i+1}(x^{(i)})$
   $\mathcal{H}_{j+1} = \mathcal{H}_j \cup \left\{(x^{(j)}, Y_{j+1}(x^{(j)}))\right\}$.
4. If the budget is exhausted then return $\widehat{x}^\star = \text{argmax}_x \bar{Y}(x)$, otherwise $j = j + 1$ and go to 2.

Clearly the key aspect is the policy $\pi(\cdot)$. Often the policy is expressed as some sort of "acquisition function" a, for instance

$$\pi(\mathcal{H}) = \text{argmax}_{x \neq \widehat{x}^\star} a(x, \widehat{x}^\star) = \text{argmax}_{x \neq \widehat{x}^\star} \mathrm{E}\left[\max\left\{0, M(x) - M(\widehat{x}^\star)\right\} \middle| \mathcal{H}\right] \quad (1)$$

which is the system design with the largest posterior expected value of improvement over the current sample best. Ideally a is chosen to learn "optimally," meaning as efficiently as possible, but the policy also has to be computable, which often means it cannot look too many steps ahead.

Gaussian processes provide a very useful framework for this sort of approach, often based on two fundamental results:

1. If $Z \sim \mathrm{N}(0,1)$ then $\mathrm{E}[\max\{0, \mu + \sigma Z\}] = \mu\Phi\left(\frac{\mu}{\sigma}\right) + \sigma\phi\left(\frac{\mu}{\sigma}\right)$ where Φ and ϕ are the cdf and density of Z, respectively.
2. If $(Z_1, Z_2) \sim \mathrm{BVN}(\mu_1, \mu_2, \sigma_1^2, \sigma_2^2, \rho)$ then $Z_1 \sim \mathrm{N}(\mu_1, \sigma_1^2)$, and

$$Z_1 | Z_2 = z \sim \mathrm{N}\underbrace{\left(\mu_1 + \rho\frac{\sigma_1}{\sigma_2}(z - \mu_2),\ \sigma_1^2(1-\rho^2)\right)}_{\text{"learning"}}.$$

The acquisition function in (1) is known as the *complete expected improvement* (CEI) policy [36]. When the posterior is the normal distribution, then using the first fundamental fact we have

$$\begin{aligned}\mathrm{CEI}(x, \widehat{x}^\star) = (m(x) - m(\widehat{x}^\star))\Phi\left(\frac{m(x) - m(\widehat{x}^\star)}{\sqrt{\mathrm{Var}(x, \widehat{x}^\star)}}\right) \\ + \sqrt{\mathrm{Var}(x, \widehat{x}^\star)}\phi\left(\frac{m(x) - m(\widehat{x}^\star)}{\sqrt{\mathrm{Var}(x, \widehat{x}^\star)}}\right)\end{aligned}$$

where $m(x) = \mathrm{E}(M(x)|\mathcal{H})$, $\mathrm{Var}(x, \widehat{x}^\star) = \mathrm{Var}(M(x) - M(\widehat{x}^\star)|\mathcal{H})$. The second fact can be exploited to compute the means and variances, conditional on $\mathcal{H}$. The CEI policy has been shown empirically to make rapid progress toward the best system.

3.3 A Convergence-Rate Perspective

Suppose that the best system is unique: $\mu(k) > \mu(k-1)$. Then as long as all the $n(x) \to \infty$, even if not all equal, we will eventually correctly select $\widehat{x}^\star = k$ due to the strong law of large numbers. But what is the best way to get to ∞? For the purposes of this section it will be useful to employ the notation $\bar{Y}_x(n(x))$ for the sample mean of $n(x)$ replications from system x, $\mu_x = \mu(x)$ and $\sigma_x = \sigma(x)$, and further to let $n(x) = \beta_x N$ where $\beta_x \geq 0$, $\sum_x \beta_x = 1$ and N is the total replication budget. The question then becomes, what choice of $\beta_1, \beta_2, \ldots, \beta_k$ makes $\lim_{N\to\infty} \Pr\{\widehat{x}^\star \neq k\}$ go to 0 the fastest?

One way to answer this question is via a large-deviation principle (LDP). Let $Z_1, Z_2, \ldots, Z_N$ be i.i.d. (μ, σ^2). If Z has finite log moment generating function, then for $z > \mu$

$$\lim_{N\to\infty} \frac{1}{N} \ln[\Pr\{\bar{Z}(N) > z\}] = -I(z)$$

where $I(\cdot)$ is a *rate function* that depends on the distribution of Z. This LDP can be interpreted as

$$\Pr\{\bar{Z}(N) > z\} \approx e^{-NI(z)} \text{ for large} N.$$

Translating to R&S, we want to choose $\beta_1, \beta_2, \ldots, \beta_k$ to maximize the smallest of the rates of decay of the pairwise probabilities of incorrect selection (PICS)

$$\text{PICS}_x = \Pr\{\bar{Y}_x(\beta_x N) - \bar{Y}_k(\beta_k N) > 0\} \approx \exp(-N I(0, \beta_x, \beta_k))$$

where $I(0, \beta_x, \beta_k)$ indicates that the rate function depends on the allocation β_x, β_k. [11] showed that if the outputs are normally distributed then the LDP rate-optimal allocation satisfies

$$\left(\frac{\beta_k}{\sigma_k}\right)^2 = \sum_{x \neq k} \left(\frac{\beta_x}{\sigma_x}\right)^2$$

$$\frac{(\mu_x - \mu_k)^2}{\frac{\sigma_x^2}{\beta_x} + \frac{\sigma_k^2}{\beta_k}} = \frac{(\mu_{x'} - \mu_k)^2}{\frac{\sigma_{x'}^2}{\beta_{x'}} + \frac{\sigma_k^2}{\beta_k}}, \quad \forall x, x' \neq k.$$

Unfortunately, this expression involves quantities that we do not know, and just plugging in estimates does not give the best possible rate (things get harder for unknown distributions because estimating LDP rates is difficult). Fortunately, [5] showed that a slight modification of the CEI policy from the previous section, called mCEI, is asymptotically equivalent to the rate-optimal allocation! This result is remarkable because CEI comes from unrelated reasoning: the Bayes-optimal allocation of the next simulation run if that run will be your last. Figure 1 illustrates the mCEI procedure's allocation in a five-system problem over 150 iterations. Notice that system $x = 4$, which is the true best, also receives the most replications.

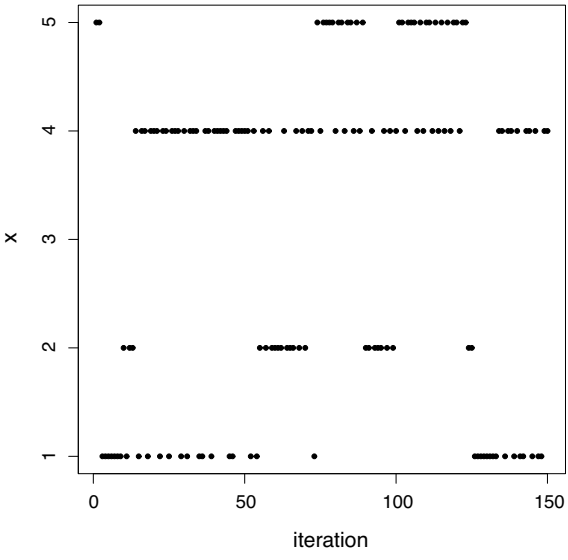


Fig. 1 Illustration of allocations from mCEI in a R&S problem with $k = 5$ systems. Each • represents a replication

Another popular policy that is in the same spirit as mCEI is optimal computer budget allocation (OCBA), which is derived through a Bayesian-inspired approximation to the posterior PCS. OCBA uses plug-in estimates and nonlinear optimization to allocate batches of replications. Although it does not achieve the rate-optimal allocation in the limit, it is quite effective empirically. See [3].

3.4 Doing Better Than "Rate Optimal"

The asymptotically optimal allocation focuses on the endgame, as sample sizes get large, and is not necessarily the best allocation for *finite* N. After all, we do not need to drive PICS to 0 to be highly confident of selecting the best. Further, in the rate-optimal allocation all $\beta_x > 0$, which means that all systems remain in play until we stop, which may imply a lot of computational overhead on each step, especially if k is large. Also, the rate-optimal allocation does not provide a way to do fixed-precision stopping. And finally, one-system-at-a-time allocation is becoming less and less attractive as it becomes easier and easier to simulate $p > 1$ systems or replications in parallel.

Often (especially when k is large) there are many bad systems we can completely eliminate from further consideration quickly. This is one way to beat rate-optimal for finite N. There are two basic strategies:

- **Screen & select**: Get a small number of replications from all system designs, create a subset $\widehat{\mathcal{S}}$ that still contains the best, then apply an efficient R&S procedure to the remainder. This usually requires splitting the α error between subset and selection so that $\Pr\{k \in \widehat{\mathcal{S}}\} \geq 1 - \alpha/2$.
- **Continuous screening**: Iteratively replicate, eliminate, replicate, eliminate and so on until one system remains. This usually requires tracking all pairwise comparisons and controling the overall error via (for instance) the Bonferroni inequality. But even for a single pairwise comparison we need results that allow "multiple looks" at the data for continuous screening.

3.4.1 Screening

We begin with a basic subset selection (screening) procedure from [30] for systems simulated independently:

Basic subset selection

1. Simulate $n(x) \geq 2$ replications from system x, set $t(x) = t_{(1-\alpha)^{\frac{1}{k-1}},\, n(x)-1}$ the $(1-\alpha)^{\frac{1}{k-1}}$ quantile of the t distribution with $n(x) - 1$ degrees of freedom, for $x = 1, 2, \ldots, k$.
2. Calculate the sample means $\bar{Y}(x)$ and sample variances

$$S^2(x) = \frac{1}{n(x)-1}\sum_{j=1}^{n(x)}\left(Y_j(x) - \bar{Y}(x)\right)^2$$

for $x = 1, 2, \ldots, k$, and also for all $x \neq x'$ compute

$$W(x, x') = \left(t(x)^2\frac{S^2(x)}{n(x)} + t(x')^2\frac{S^2(x')}{n(x')}\right)^{1/2}.$$

3. Form the subset

$$\widehat{\mathcal{S}} = \left\{x\colon \bar{Y}(x) \geq \bar{Y}(x') - W(x, x') \text{ for all } x' \neq x\right\}.$$

The following reasoning is behind many subset selection procedures:

$$\begin{aligned}
&\Pr\{k \in \widehat{\mathcal{S}}\} \\
&\quad = \Pr\left\{\bar{Y}(k) \geq \bar{Y}(x) - W(k, x),\ x \neq k\right\} \\
&\quad = \Pr\left\{\bar{Y}(k) - \bar{Y}(x) - [\mu(k) - \mu(x)] \geq -W(k, x) - [\mu(k) - \mu(x)],\ x \neq k\right\} \\
&\quad \geq \Pr\left\{\bar{Y}(k) - \bar{Y}(x) - [\mu(k) - \mu(x)] \geq -W(k, x),\ x \neq k\right\}.
\end{aligned}$$

Notice that the statistic $\bar{Y}(x) - \bar{Y}(x') - [\mu(x) - \mu(x')]$ has mean 0 for all $x \neq x'$, allowing the $W(x, x')$'s to be derived to give the desired probability based only on their variances. The survivors of subset selection can then be passed on to something like an IZ R&S procedure; see for instance, [30].

3.4.2 Fully Sequential Screening

The downside of using subset selection for screening, then applying IZ R&S to the survivors to select the best, is that the effectiveness of subset selection depends on the choice of sample size $n(x)$, and a good choice of $n(x)$ depends on the true means and variances of the outputs, which are unknown. A natural generalization is to do many rounds of subset selection, perhaps only stopping when there is one system remaining. Fully sequential, eliminating procedures do just that. Many such procedures are built on modeling the simulation output process as Brownian motion, a continuous-time, continuous-state stochastic process we review next.

Let $\{\mathcal{B}(t);\ t \geq 0\}$ be *standard Brownian motion (BM)*. Then

1. $\mathcal{B}(0) = 0$.
2. $\{\mathcal{B}(t);\ t \geq 0\}$ is almost surely continuous.
3. $\{\mathcal{B}(t);\ t \geq 0\}$ has independent increments: $\mathcal{B}(t) \perp \mathcal{B}(t+s) - \mathcal{B}(t)$.
4. $\mathcal{B}(t) - \mathcal{B}(s) \sim \mathrm{N}(0, t-s),\ 0 \leq s \leq t$.

An important generalization is BM with *drift* δt defined as $\mathcal{B}(t;\delta) = \mathcal{B}(t) + \delta t$. Therefore, $\sigma\mathcal{B}(t;\delta/\sigma) = \sigma\mathcal{B}(t) + \delta t$ has drift δt and variance $\sigma^2 t$. The relationship between BM with drift and R&S is as follows: Consider the sum of pairwise differences between the best system k and some other system x: $D_x(r) = \sum_{j=1}^{r}(Y_j(k) - Y_j(x))$, with $\sigma_{kx}^2 = \mathrm{Var}(Y_j(k) - Y_j(x))$, $\delta_{kx} = \mu(k) - \mu(x)$, and all outputs normally distributed. Then

$$\{D_x(r);\ r = 1, 2, \ldots\} \stackrel{\mathcal{D}}{=} \{\sigma_{kx}\mathcal{B}(r;\delta_{kx}/\sigma_{kx});\ r = 1, 2, \ldots\}\,. \tag{2}$$

That is, we can represent the cumulative pairwise-differences of two systems' outputs (one being the best) as scaled Brownian motion with positive drift but monitored only at integer times. The following fundamental result relates the crossing times and probabilities of Brownian motion observed continuously, and only at integer times:

Theorem 1 ([17]) *Suppose $\delta > 0$, and we have a continuous function $g(t) \geq 0$ for all $t \geq 0$. Let*

$$T_d = \min\{r\colon |\mathcal{B}(r;\delta)| \geq g(r),\ r = 1, 2, \ldots\}$$
$$T_c = \min\{t\colon |\mathcal{B}(t;\delta)| \geq g(t),\ t \geq 0\}.$$

Then $T_c \leq T_d$ a.s. and $\Pr\{\mathcal{B}(T_d;\delta) \leq -g(T_d)\} \leq \Pr\{\mathcal{B}(T_c;\delta) \leq -g(T_c)\}$.

Thus, if crossing $-g(t)$ is an undesirable event—such as causing us to eliminate the true best system—then such an event is even less likely if we only observe the process at integer times. A lot is known about the probability of BM crossing boundaries of the form $\pm g(t)$. This, along with Theorem 1 facilitates designing regions that control the probability of a selection error.

The relationship in (2) applies to synchronized, pairwise differences. Reference [14] noted that the BM model can also extend to unequal samples sizes on a non-integer time scale via

$$\left[\frac{\sigma^2(k)}{n(k)} + \frac{\sigma^2(x)}{n(x)}\right]^{-1}\left[\bar{Y}(k) - \bar{Y}(x)\right] \stackrel{\mathcal{D}}{=} \mathcal{B}\left(\left[\frac{\sigma^2(k)}{n(k)} + \frac{\sigma^2(x)}{n(x)}\right]^{-1};\ \mu(k) - \mu(x)\right). \tag{3}$$

Illustration: Paulson's Procedure

Because fully sequential, eliminating procedures have been so important in R&S we take a deep dive into Paulson's Procedure [32], a fully sequential IZ procedure for known, common variance.

Paulson's Procedure

0. Set $\mathcal{S} = \{1, 2, \ldots, k\}$, choose $\lambda \in (0, \delta)$, set $a = \frac{\sigma^2}{\delta - \lambda}\ln\left(\frac{k-1}{\alpha}\right)$ and set $r = 0$.
1. Set $r = r + 1$. Simulate $Y_r(x)$, $\forall x \in \mathcal{S}$.
2. Mark systems $\ell \in \mathcal{S}$ for elimination if

$$\min_{x \in \mathcal{S}} \left\{ \sum_{j=1}^{r} (Y_j(\ell) - Y_j(x)) \right\} < \min\{0, -a + \lambda r\}.$$

3. Remove all marked systems from $\mathcal{S}$.
4. If $|\mathcal{S}| = 1$ then stop and select system $\mathcal{S}$ as best; else go to Step 1.

Paulson's procedure tries to be *observation efficient* by attempting to eliminate systems after *each* additional replication. Notice that elimination decisions are highly coordinated, and require looking at $\binom{|\mathcal{S}|}{2}$ pairwise differences. Paulson guarantees $\Pr\{\text{select } k \mid \mu(k) - \mu(k-1) \geq \delta\} \geq 1 - \alpha$, but the guarantee is not clear when there are systems closer than δ. The extension to unknown and unequal variances is not hard; as an illustration the case of unknown common variance σ^2 will be presented later. The procedure ends by or before step $N + 1 = \lfloor a/\lambda \rfloor + 1$.

The large-deviation result supporting Paulson is as follows:

Theorem 2 *Suppose $Z_1, Z_2, \ldots$ are i.i.d. $\mathrm{N}(\Delta, \sigma^2)$ with $\Delta < 0$. Then for any constant $a > 0$*

$$\Pr\left\{ \sum_{j=1}^{r} Z_j > a \text{ for some } r < \infty \right\} \leq \exp\left(\frac{2\Delta a}{\sigma^2} \right).$$

Notice that since $\Delta < 0$ we expect the sum to drift *down*; this large deviation result bounds the probability it drifts *up* more than a. In the IZ formulation, we believe that $Y_j(x) - Y_j(k)$ has negative drift of at least $-\delta$ for all $x \neq k$. Attacking the pairwise differences we would like to choose a to obtain

$$\Pr\{k\text{eliminated}\} \leq \sum_{x=1}^{k-1} \Pr\{x \text{ eliminates } k\} = \sum_{x=1}^{k-1} \Pr\{\mathrm{ICS}_x\} \leq \alpha.$$

Proof. We consider the probability that system $x \neq k$ incorrectly eliminates system k in isolation.

$$\begin{aligned}
\Pr\{\mathrm{ICS}_x\} &\leq \Pr\left\{ \sum_{j=1}^{r} (Y_j(k) - Y_j(x)) < -a + \lambda r \text{ some } r \leq N + 1 \right\} \\
&= \Pr\left\{ \sum_{j=1}^{r} (Y_j(x) - Y_j(k) + \lambda) > a \text{ some } r \leq N + 1 \right\} \\
&\leq \Pr\left\{ \sum_{j=1}^{r} (Y_j(x) - Y_j(k) + \lambda) > a \text{ some } r < \infty \right\} \\
&\leq \exp\left(\frac{2(\mu(x) - \mu(k) + \lambda)a}{2\sigma^2} \right) \leq \exp\left(\frac{(-\delta + \lambda)a}{\sigma^2} \right) = \frac{\alpha}{k-1}
\end{aligned}$$

where the last step follows because we set $a = \frac{\sigma^2}{\delta - \lambda} \ln\left(\frac{k-1}{\alpha}\right)$. A common choice for the slope is $\lambda = \delta/2$. □

There are a number of ways of improving on Paulson's Procedure, including (a) accomodating unknown and unequal variances (see Sect. 3.7); (b) exploiting tighter Brownian-motion large-deviation results (notice the result we used protected system k for all $r < \infty$; see [19]); (c) facilitating variance-dependent sampling so that systems with low variance need to be simulated less (see [14]); (d) providing a PGS guarantee for when $\mu(k) - \mu(k-1) < \delta$ (see Sect. 3.6); (e) avoiding breaking up into paired comparisons and using Bonferroni's inequality (see [6]); and (f) exploiting common random numbers (see Sect. 3.5).

3.5 *Common Random Numbers*

R&S procedures that employ pairwise comparisons can often be "sharpened" by using common random numbers (CRN) because

$$\mathrm{Var}(Y(x) - Y(x')) = \mathrm{Var}(Y(x)) + \mathrm{Var}(Y(x')) - 2\,\mathrm{Cov}(Y(x), Y(x'))$$

and CRN tends to make $\mathrm{Cov}(Y(x), Y(x')) > 0$ [29]. However, to fully realize the CRN effect requires $n(x) = n(x')$ so that replications can be paired.

As an illustration, the impact of CRN on subset selection (Sect. 3.4.1) is that

$$W(x, x') = \left(t(x)^2 \frac{S^2(x)}{n(x)} + t(x')^2 \frac{S^2(x')}{n(x')}\right)^{1/2}$$

becomes

$$W(x, x') = \left(t^2 \frac{S^2(x, x')}{n}\right)^{1/2}$$

where $S^2(x, x') = S^2(x) + S^2(x') - 2\widehat{\mathrm{Cov}}(x, x')$ and $\widehat{\mathrm{Cov}}(x, x')$ is the estimated covariance. Thus, positive covariance should make it more difficult for inferior systems to remain in the subset because the boundary is tighter. Similarly, the impact on Paulson's Procedure (Sect. 3.4.2) with equal, known variance σ^2 and CRN-induced correlation $\rho > 0$ is that the elimination boundary has intercept

$$a = \frac{\sigma^2(1-\rho)}{\delta - \lambda} \ln\left(\frac{k-1}{\alpha}\right) \text{ rather than } a = \frac{\sigma^2}{\delta - \lambda} \ln\left(\frac{k-1}{\alpha}\right).$$

Again, positive covariance should make it more difficult for inferior systems to remain in the subset because the elimination boundary is narrower.

Simulation languages have random number "streams" that map to starting seeds that are *very* far apart; therefore, we can assign a unique stream to each random process and replication to enhance the impact of CRN [22, 29].

3.6 "Good Selection"

The IZ-PCS paradigm $\text{PCS} = \Pr\{\widehat{x}^\star = k \mid \mu(k) - \mu(k-1) \geq \delta\} \geq 1 - \alpha$ has been the most widely adopted in practice. Typically, δ is chosen as the "smallest practically significant difference," which may not be close to the *actual* differences $\mu(k) - \mu(x)$. In fact when k is large we expect several "good" systems, and very many inferior ones. Thus, guaranteed probability of good selection

$$\text{PGS} = \Pr\left\{\mu(k) - \mu(\widehat{x}^\star) < \delta\right\} \geq 1 - \alpha$$

is more meaningful than PCS because it can be interpreted as an acceptable bound on the optimality gap.

Empirical experience suggests that procedures with an IZ-PCS guarantee also provide a PGS guarantee; however, counterexamples can be created. IZ procedures *without elimination* (e.g., Rinott) can often be shown to guarantee PGS, but elimination makes proving PGS difficult. An excellent comprehensive reference is [7]. A condition that insures both PCS and PGS is stated in the following theorem:

Theorem 3 ([28]) *Suppose a R&S procedure creates* $\widehat{\mu}(1), \widehat{\mu}(2), \ldots, \widehat{\mu}(k)$ *that guarantee* $\Pr\{\widehat{\mu}(k) > \widehat{\mu}(i),\ \forall i \neq k \mid \mu(k) - \mu(k-1) \geq \delta\} \geq 1 - \alpha$. *Then if*

$$\begin{pmatrix} \widehat{\mu}(k) \\ \widehat{\mu}(k-1) - \mu(k-1) + (\mu(k) - \delta) \\ \vdots \\ \widehat{\mu}(1) - \mu(1) + (\mu(k) - \delta) \end{pmatrix}$$

has the same distribution as estimators would have had in the corresponding slippage configure problem, then the procedure also guarantees $\text{PGS} \geq 1 - \alpha$.

Normally distributed output procedures like Rinott that do not adapt to the sample means often satisfy the conditions of this theorem. Unfortunately, lack of adaptation also tends to lead to inefficiency.

Reference [37] make an adjustment to Paulson's procedure so that it provides a good-selection guarantee. Recall that Paulson eliminates system ℓ if for some other system x we have $\sum_{j=1}^{r}(Y_j(\ell) - Y_j(x)) < -a + \lambda r$. Instead, [37] use the condition $\sum_{j=1}^{r}(Y_j(\ell) - Y_j(x) + \delta) < -a + \lambda r$. Notice that when $\mu(k) - \mu(\ell) < \delta$, the sum of differences $\sum_{j=1}^{r}(Y_j(\ell) - Y_j(k) + \delta)$ still has positive drift. Thus, good systems should survive to the end, where [37] then select the sample-best system.

A Bayesian "good selection" R&S procedure stops when it has collected enough output so that there is a system $\widehat{x}^\star$ for which

$$\Pr\{M(\widehat{x}^\star) > M(x) - \delta,\ \forall x \neq \widehat{x}^\star \mid \mathcal{H}\} \geq 1 - \alpha.$$

This is computable under some assumptions, but if not then it can be approximated or bounded. The interpretation is that "With probability at least $1 - \alpha$ the *random problem* from your space of priors is one for which the *fixed system* $\widehat{x}^\star$ is good." This contrasts with the frequentist perspective: The *random system* $\widehat{x}^\star$ chosen by the procedure has probability at least $1 - \alpha$ of being good for this *fixed problem* [7].

3.7 Unknown Variances

As a general rule, neither known nor equal variances can be assumed in simulation R&S problems. For procedures that break into pairwise differences the variance of each pairwise difference can be estimated separately, which is also helpful for using CRN.

A useful result that sits behind many R&S procedures is this: If $Z_1, Z_2, \ldots, Z_{n_0}$ are i.i.d. $\mathrm{N}(\mu, \sigma^2)$ then $\bar{Z}$ is independent of S^2. Thus, using a "first-stage" S^2 to calibrate the additional simulation needed does not introduce bias. If done cleverly, we can derive the PCS *conditional* on S^2 and then uncondition. Not surprisingly, using estimated σ^2 increases E(sample size) relative to known variance [26].

Illustration: Unknown Common Variance Paulson Recall in Paulson that we set $\lambda = \delta/2$ and $a = \dfrac{2\sigma^2}{\delta} \ln\left(\dfrac{k-1}{\alpha}\right)$, assuming σ^2 was known. Suppose we estimate σ^2 from an initial simulation of n_0 replications from each system by

$$S^2 = \frac{1}{k(n_0 - 1)} \sum_{x=1}^{k} \sum_{j=1}^{n_0} (Y_j(x) - \bar{Y}(x))^2.$$

We will exploit two useful facts:

$$\frac{k(n_0 - 1)S^2}{\sigma^2} \sim \chi^2_d \text{ with } d = k(n_0 - 1) \text{ and } \mathrm{E}\left[\exp(t\chi^2_d)\right] = (1 - 2t)^{-d/2}$$

when χ^2_d is a chi-squared random variable with d degrees of freedom. The approach we take is to set $a = \eta S^2/\delta$ and see what η needs to be to get the desired PCS.

In the proof of Paulson's procedure we used a large-deviation result to show that for *fixed* a and $\lambda = \delta/2$

$$\Pr\{\mathrm{ICS}_x\} \leq \exp\left(-\frac{\delta}{2\sigma^2} a\right).$$

With $a = \eta S^2/\delta$, to obtain $\Pr\{\text{ICS}_x\} \le \alpha/(k-1)$ we need η to satisfy

$$\begin{aligned}\Pr\{\text{ICS}_x\} &= \mathrm{E}\left[\Pr\{\text{ICS}_x \mid S^2\}\right] \le \mathrm{E}\left[\exp\left(-\frac{\delta}{2\sigma^2}\frac{\eta S^2}{\delta}\right)\right] \\ &= \mathrm{E}\left[\exp\left(\underbrace{-\frac{\eta}{2d}}_{t}\underbrace{\frac{dS^2}{\sigma^2}}_{\chi^2_d}\right)\right] = \left(1-\frac{-2\eta}{2d}\right)^{-d/2} = \frac{\alpha}{k-1}.\end{aligned}$$

Solving for η gives

$$\eta = \left(\frac{\alpha}{k-1}\right)^{-2/d} - 1.$$

Notice that the independence of $\bar{Y}$ and S^2 is critical.

Paulson is great for illustrating concepts, but the limitation to equal variances and no common random numbers makes it rarely used in simulation. There are many descendants, with one of the most statistically efficient and robust being KN [19], which uses a tighter Brownian motion result; allows unequal variances and CRN; has been shown to be asymptotically valid for non-normal output data (discussed below); and has been implemented in commercial simulation languages and in parallel.

3.8 A Note on Asymptotic Analysis

Asymptotic analysis of R&S procedures is useful in at least three contexts:

1. Establishing that a procedure will work when core assumptions such as normality are violated (typically as $\delta \to 0$ in a way that also makes the problem harder).
2. Comparing the efficiency of procedures that are difficult to evaluate in finite samples (typically as $1-\alpha \to 1$ so that behavior becomes deterministic).
3. Comparing the efficiency of procedures with estimated variances relative to their known-variance counterparts (typically as $\delta \to 0$ drives $n_0 \to \infty$).

Setting 1 helps explain why normal-theory IZ procedures seem to work well more generally, while Setting 2 is often the only way (other than empirically) to compare procedures that eliminate systems.

For Setting 1 a *meaningful* limit is essential: If $\mu(k) - \mu(x)$ is fixed, then as we let $\delta \to 0$ for procedures with sample size proportional to $1/\delta^2$, we have PCS $\to 1$ for almost any kind of data by the strong law of large numbers. Reference [20] let $\mu(k) = \mu$ and $\mu(x) = \mu - \delta$ for $x \ne k$. Notice that as $\delta \to 0$ the sample size goes to ∞ but the problem itself also gets harder. Is this a relevant setting? If $\delta \gg \mu(k) - \mu(x)$ then any system design is acceptable. If $\delta \ll \mu(k) - \mu(x)$ then a procedure will tend to simulate so many replications that it will select the best. Thus $\mu(x) = \mu(k) - \delta$ is the critical regime.

For R&S procedures based on Brownian motion, a key tool for asymptotic analysis via Setting 1 is

Theorem 4 (Donsker's Theorem) *If* $Y_1, Y_2, \ldots$ *are i.i.d.* (μ, σ^2) *with* $\sigma^2 < \infty$ *then as* $N \to \infty$

$$\frac{\sum_{j=1}^{\lfloor Nt \rfloor} Y_j - Nt\mu}{\sigma\sqrt{N}} \xrightarrow{\mathcal{D}} \mathcal{B}(t), \ 0 \le t \le 1.$$

The usual Central Limit Theorem drops out at $t = 1$. Donsker's Theorem goes further, stating that very general i.i.d. output processes, standardized the right way, look like Brownian motion as we get more and more data. In many IZ R&S procedures we can take $Y_j = (Y_j(x) - Y_j(x'))$, and letting $\delta \to 0$ drives the sample size to ∞ when $N \propto 1/\delta^2$.

4 Parallel R&S

The future of simulation, and certainly simulation optimization, is parallel computing. Simulation languages have already been redesigned to run in the cloud, where computer time is "rented." For instance, the commercial product Simio automatically exploits multi-core/multi-thread personal computers, and its portal version can recruit up to 10,000 processors from Microsoft Azure to run simulations in parallel.

The availability of cheap, easy-to-use parallel computing greatly extends the R&S limit in terms of problem size, k. However, since one may have to pay for the service, the focus of "efficiency" in R&S shifts from being observation-efficient to being computationally efficient as measured by wall-clock or rental time. Thus, it may be acceptable to waste simulation-generated replications to avoid idling processors and get the R&S problem solved faster. Factors such as heterogeneous processors, communication delays, processor failures, etc. that may disrupt the usual synchronization in R&S procedures now become relevant.

To describe parallel R&S we assume a master-worker paradigm: 1 Master process performing calculations and generating new jobs, and p Worker processes executing simulation and calculation jobs. While not the only possible architecture, it is a common one. The following framework for parallel R&S is based on [15].

We represent a R&S procedure as a sequence of jobs generated by the Master, $\mathcal{J} = \{J_j : 1 \le j \le M\}$, where Job j is an ordered list

$$J_j \equiv \{\underbrace{(\mathcal{Q}_j, \Delta_j, \mathcal{U}_j)}_{\text{simulate}}, \underbrace{(\mathcal{P}_j, \mathcal{C}_j)}_{\text{calculate}}\}.$$

The components of job j are

- $\mathcal{Q}_j \subseteq \{1, 2, \ldots, k\}$ indices of systems to be simulated;
- $\Delta_j = \{\Delta_{xj}\}$ how many replications to take from each system $x \in \mathcal{Q}_j$;

- $\mathcal{U}_j$ (optional) the assigned block of random numbers;
- $\mathcal{C}_j$ the list of non-simulation calculations or operations to perform; and
- $\mathcal{P}_j$ the list of jobs that must complete before executing the calculation $\mathcal{C}_j$.

Using this computational paradigm, most of the (non-parallel) R&S procedures presented so far look something like the Nominal R&S Procedure below.

Nominal R&S Procedure

1. Until fixed-precision or fixed-budget ending condition reached, do
2. For $\ell = 1, 2, \ldots$

   a. Execute simulation jobs for non-eliminated or active systems:

   $$J_\ell = [\{(\text{system } i, 1 \text{ rep}), (\emptyset)\}, \ldots, \{(\text{system } j, 1 \text{ rep}), (\emptyset)\}, \ldots]$$

   b. Execute a comparison job using the results from Step 2a.

   $$J'_\ell = \{(\emptyset), (\text{all jobs in } J_\ell, \mathcal{C}_\ell)\}$$

   where $\mathcal{C}_\ell$ performs calculations on all non-eliminated or active systems.

The nominal procedure enforces many of the assumptions necessary for both small-sample and asymptotic analysis by "synchronized coupling." To directly parallelize it, the Master could spread job J_ℓ out among the p workers, but then many workers may be idle while waiting for the coupled Step 2b to complete. Such issues do not arise in a single-processor setting.

Why not just use the outputs from the simulation jobs as soon as they complete, rather than waiting? [24] address this question, and shows that new statistical issues arise. Recall there are $p + 1$ processors, consisting of 1 Master and p Workers, and suppose that all simulation jobs are assigned by the Master to a Worker in round robin fashion as follows: system $1, 2, \ldots, k, 1, 2, \ldots$. An eliminated system is removed from the remaining list. Let $Z_j(x)$ be the *input sequence*—the result of jth replication from alternative x *requested* by the Master—with execution and communication time $T_j(x)$. Similarly, let $Y_j(x)$ be the corresponding *output sequence*, meaning the jth output from alternative x *returned* to the Master. If the R&S procedure uses each output as soon as it is available to the Master, then the following new statistical issues arise.

1. The procedure is working with random sample sizes at each comparison step, rather than prescribed numbers of replications.
2. The $Y_j(x)$, $j = 1, 2, \ldots$ are not i.i.d. To see this, suppose $k = 1$, $Z_j(x) = T_j(x) \sim \text{Expon}(\mu(x))$. Then it can be shown that $\text{E}(Y_j(x)) = \mu(x)\left(1 - \left(1 - \frac{1}{p}\right)^j\right)$ because jobs with short execution times return to the master sooner.
3. There is a subtle dependence among systems' outputs caused by elimination of some systems impacting the number of replications of other systems.

Thus, parallelization takes some careful thought, not only from a computer science point of view, but also with respect to the statistical validity of the R&S procedure.

4.1 New Measures of Efficiency

How do we define "efficiency" in this new parallel paradigm?

- Let $0 < T_j < \infty$ be the wall-clock time Job J_j finishes, so that the ending time of the procedure is $T_e(\mathcal{J}) = \max_{j=1,2,\ldots,M} T_j$.
- Let $c(p, s)$ be the cost to purchase p processors for s time units.

With these definitions we can define revised "efficient" objectives.

Fixed Precision:

Achieve a statistical guarantee while being cost efficient:

$$\begin{aligned} \text{minimize}_{p,\mathcal{J}} \quad & \mathrm{E}[\beta_t \underbrace{T_e(\mathcal{J})}_{\text{time}} + \beta_c \underbrace{c(p, T_e(\mathcal{J}))}_{\text{cost}}] \\ \text{s.t.} \quad & \Pr\{\underbrace{G(\widehat{x}^\star, k)}_{\text{good event}}\} \geq 1 - \alpha. \end{aligned}$$

Fixed Budget: Minimize a loss for the selected system within a budget:

$$\begin{aligned} \text{minimize}_{p,\mathcal{J}} \quad & \mathrm{E}[\underbrace{\mathcal{L}(G^c(\widehat{x}^\star, k), \mathcal{J})}_{\text{loss from bad event}}] \\ \text{s.t.} \quad & \underbrace{c(p, T_e(\mathcal{J}))}_{\text{cost}} \leq b. \end{aligned}$$

Notice that for both fixed-precision and fixed-budget formulations, the decision variables are the number of processors to rent p and the jobs to execute $\mathcal{J}$. For fixed precision it is possible that we would only have one of β_t or β_c to be non-zero, depending on whether the time to reach a decision or the rental cost to reach a decision is most important.

To the best of our knowledge no R&S procedure has yet been created that directly attacks one of these formulations. Table 1 cites much of the existing literature on parallel R&S, divided into fixed-budget vs. fixed-precision, and load balancing to enforce standard non-parallel assumptions vs. a uniquely parallel paradigm. Consider for instance [24], and its "phantom clock." The underlying procedure is KN, a fully sequential procedure that uses pairwise sums of differences. Reference [24] note that even when the input and output sequences are not the same, if the procedure only makes comparisons at times t when $\sum_{j=1}^{t}(Y_j(k) - Y_j(x)) = \sum_{j=1}^{t}(Z_j(k) - Z_j(x))$

Table 1 Selected parallel R&S literature

R&S Procedure	Load balancing (Standard assumptions)	Comparison timing (Relaxed assumptions)
Fixed-precision	Simple divide and conquer [4]	
	Vector-filling procedure [24]	Asymptotic parallel selection [24]
	Good selection procedure [31]	bi-PASS [34]
	Strategic updating [38]	
Fixed-budget	Parallel OCBA [25]	
	Asynchronous OCBA/KG [18]	bi-PASS [34]

then the order of return from the workers does not matter. Strictly enforcing this is load balancing. Instead, [24] insert a "phantom job" at the end of each round-robin job cycle $(1, 2, \ldots, k, \text{ph}, 1, 2, \ldots, k, \text{ph}, \ldots)$, and then only compare systems when a phantom job returns to the Master. They show that by doing this the sums of differences are only out of sync by an asymptotically negligible amount.

4.2 New Objectives

Does insuring a prespecified PCS or PGS continue to make sense if k is very large? For instance, if $k > 1{,}000{,}000$ systems, is it sensible or even computationally feasible to identify the single best or near-best with high probability? In such a problem we expect many bad systems, but also a lot of good ones. Trying to achieve PCS or PGS, which are family-wide statements, in such a setting runs counter to approaches in large-scale statistical inference of controlling "error rates" [8]. Specifically, to control PCS requires more and more effort per system as k increases. As we argue below, rates such as "false discovery" can be attained with little or no "k effect."

But why apply R&S for such large-k problems anyway? Surely so many systems arise from combinations of more basic decision variables, which suggests using a search algorithm rather than exhaustive simulation. However, from a practical perspective, the key is to *actually solve the problem* in some effective way. As discussed earlier, R&S is the only SO technique that can control all sources of error. Therefore, if parallel computing extends the R&S limit enough to encompass a problem, it makes sense to use it.

This motivates consideration of new goals for parallel R&S:

- More scalable—but still useful and understandable—error control than PCS or PGS. As an example we discuss *Expected False Elimination Rate (EFER)*, which is the fraction of good systems eliminated.
- Avoid procedures with coupled operations and synchronization, to facilitate parallelization. As an example we describe parallel adaptive survivor selection (PASS), and a specific instance bisection-PASS (bi-PASS).

For some known constant $\mu^\star$, that we call the "standard," let $S_x(n) = \sum_{j=1}^n (Y_j(x) - \mu^\star) = \sum_{j=1}^n Y_j(x) - n\mu^\star$. Suppose we have a non-decreasing function $g_{x,\alpha}(\cdot) \geq 0$ with the property that

$$\Pr\{S_x(n) \leq -g_{x,\alpha}(n), \text{ some } n < \infty\} \begin{cases} \leq \alpha, & \mu(x) \geq \mu^\star \\ = 1, & \mu(x) < \mu^\star. \end{cases}$$

Finally, let $\mathcal{G} = \{x : \mu(x) \geq \mu^\star\}$, which we refer to as the "good systems." Consider the following parallel procedure:

Parallel Survivor Selection (PSS)

1. Given a standard $\mu^\star$, an increment Δn and a budget.
2. Let $\mathsf{W} = \{1, 2, \ldots, p\}$ be the set of available workers; $\mathcal{Q} = \{1, 2, \ldots, k\}$ the set of surviving systems; and $n(x) = 0$ for all $x \in \mathcal{Q}$.
3. Until the budget is consumed, do
   a. While there is an available worker in W, do in parallel:
      i. Remove a system $x \in \mathcal{Q}$ and assign to available worker $w \in \mathsf{W}$
      ii. $j = 1$
      iii. while $j \leq \Delta n$
      Simulate $Y_{n(x)+j}(x)$. If $S_x(n(x) + j) \leq -g_{x,\alpha}(n(x) + j)$ then eliminate system x and break loop. Else $j = j + 1$.
      iv. If x not eliminated then return to $\mathcal{Q} = \mathcal{Q} \cup \{x\}$ and $n(x) = n(x) + \Delta n$.
      v. Release worker w to available workers W.
4. Return $\mathcal{Q}$.

Reference [33] show that EFER $= \mathrm{E}[|\mathcal{G} \cap \mathcal{Q}^c|]/|\mathcal{G}| \leq \alpha$. That is, the expected fraction of good systems eliminated by the procedure is no greater than α. Critically, the function g depends only on x and α, but *not* k.

The generic boundary function $g(\cdot)$ needs to insure that driftless Brownian motion ($\mu(x) = \mu^\star$) crosses with probability no more than the EFER α, while Brownian motion with negative drift ($\mu(x) < \mu^\star$) crosses with probability 1. Reference [9] note that driftless Brownian motion grows to ∞ at rate $O(\sqrt{t \log\log(t)})$, while BM with negative drift goes to $-\infty$ at rate $O(t)$. Thus $g(\cdot)$ needs to be between these two; they suggest $g(t) = \sqrt{[c + \log(t + 1)](t + 1)}$, then tuning c to get the desired error control α. The function $g_{x,\alpha}$ also includes time scaling by σ_x^2 or an estimate of it.

PSS requires no coupling and keeps the workers constantly busy; it could perhaps be made more efficient by making Δn depend on the system. Of course, the decoupling occurs because the standard $\mu^\star$ is known. However, the EFER is still controlled at $\leq \alpha$, and elimination still occurs with probability 1, if we replace $\mu^\star$ by $\mu^\star(n) \leq \mu^\star$ where $\mu^\star(n) \uparrow \mu^\star$, because a system eliminated by a smaller standard would also have been eliminated by a larger standard, and a system protected from a larger standard would also be protected from a smaller one. This suggests trying

to *learn* a standard that achieves our objectives empirically, which is called Parallel *Adaptive* Survivor Selection.

Generically, we define the standard to be $\mu^\star = s(\mu_1, \mu_2, \ldots, \mu_k, \mu^+)$. Some examples of possibly interesting standards are

- Protect the best or ties: $\mu^\star = \mu_k$.
- Protect the top m: $\mu^\star = \mu_{k-m+1}$.
- Protect the best and everything as good as a known μ^+: $\mu^\star = \min\{\mu^+, \mu_k\}$.

The key is to *learn* the standard's value in a way that still avoids coupling and does not affect the EFER. bi-PASS uses the standard

$$\bar{\mu} = \frac{1}{|\mathcal{Q}|} \sum_{x \in \mathcal{Q}} \bar{Y}(x)$$

the average of the sample means of the current survivors. Thus, the standard acts like a bisection search. Under some conditions it can be shown that the EFER is still $\leq \alpha$. Notice also that updating $\bar{\mu}$ is fast for the Master, and can occur whenever replications are returned from the Workers.

4.3 Parting Thoughts

Computer science issues really matter in parallel R&S: There is not one, unique parallel architecture, and customizations can be valuable. Message passing via MPI is conceptually easy, but unexpected behavior can occur, and passing messages does take time. Processors may be heterogeneous, and results can be lost. Memory may be shared or not. The overhead to load a simulation onto a processor can be substantial, so one also needs to consider the fixed cost to set up a simulation, as well as marginal time per replication. And management of pseudorandom numbers can be tricky, e.g., if we want to use CRN.

However, when a simulation optimization problem can be treated as a R&S problem then it can be "solved" and all three errors can be controlled. High-performance, parallel computing extends the "R&S limit" but introduces new statistical and computational problems. Standard assumptions may be violated, and "cost" no longer equals the number of replications.

5 Other Formulations

Although our focus has been on the best-mean problem with normally distributed outputs, the R&S literature is much broader. Many R&S procedures have been created for specific non-normal data; e.g., Poisson. R&S procedures have also been created

for other performance measures; e.g., probabilities and quantiles. Selecting the system that is *most likely* to be the best is called "multinomial selection," which may make sense for one-shot decisions. Selecting the best system better than a *standard* (either system or constant value) has also received attention. See [1].

The Holy Grail is a R&S procedure that works for virtually any performance measure (mean, probability, quantile) and output data distribution (normal, non-normal). Let $\theta(x)$ be the generic performance measure, and $\widehat{\theta}(x)$ a point estimator. Two insights make an omnibus procedure possible:

1. If we can construct estimators $\widehat{\theta}(x)$ of parameters $\theta(x)$ such that

$$\Pr\left\{\widehat{\theta}(x) - \widehat{\theta}(k) - (\theta(x) - \theta(k)) \le \delta,\ \forall x \ne k\right\} \ge 1 - \alpha \tag{4}$$

   then

$$\text{PGS} = \Pr\{\theta(k) - \theta(\widehat{x}^\star) \le \delta\} \ge 1 - \alpha.$$

2. Given a sample of output data, we can estimate the achieved probability in (4) using *bootstrapping*, and then increase the sample size until it is $\ge 1 - \alpha$.

Suppose we have N replications from each of the k systems, and let $\widehat{x}^\star = \text{argmax}_x \widehat{\theta}(x)$, the sample best. Then the bootstrap estimate of PGS based on B bootstrap samples is

$$\widehat{\text{PGS}} = \frac{1}{B}\sum_{b=1}^{B} \prod_{x \ne \widehat{x}^\star} \mathcal{I}\left\{\widehat{\theta}^{(b)}(x) - \widehat{\theta}^{(b)}(\widehat{x}^\star) - \left[\widehat{\theta}(x) - \widehat{\theta}(\widehat{x}^\star)\right] \le \delta\right\}$$

where $\widehat{\theta}^{(b)}(x)$ comes from independent bootstrap samples of size N and $\mathcal{I}(\cdot)$ is the indicator function. The omnibus procedure is to increase N (generate more simulation output) until this bootstrap estimate is $\ge 1 - \alpha$. Reference [23] showed this approach to be asymptotically valid under very mild conditions on the data as $\delta \to 0$.

As an illustration suppose $\theta(x)$ is the mean.

Simulation output: $[Y_1(x), \ldots, Y_N(x)] \to \bar{Y}(x), \quad x = 1, 2, \ldots, k,$ with $\widehat{x}^\star = \text{argmax}_x \bar{Y}(x)$ the current sample best with N replications.

Bootstrap: Resample the simulation outputs B times with replacement to get $\left[Y_1^{(b)}(x), \ldots, Y_N^{(b)}(x)\right] \to \bar{Y}^{(b)}(x),\ x = 1, 2, \ldots, k, b = 1, 2, \ldots, B.$

Estimate PGS:

$$\widehat{\text{PGS}} = \frac{1}{B}\sum_{b=1}^{B} \prod_{x \ne \widehat{x}^\star} \mathcal{I}\left\{\bar{Y}^{(b)}(x) - \bar{Y}^{(b)}(\widehat{x}^\star) - \left[\bar{Y}(x) - \bar{Y}(\widehat{x}^\star)\right] \le \delta\right\}.$$

Notice that we can incorporate CRN by bootstrapping *vectors* of replications, where each vector contains one replication from each of the k systems generated using CRN.

6 Multi-armed Bandits

"Multi-armed bandit" is a slang name for slot machines. To play a slot machine one inserts a coin or token and pulls the machine's mechanical "arm." Typically the coin is lost (negative reward), but occasionally the machine "pays out" a positive reward. A slot machine player would like to find the machine among many with the highest payout while losing as little money as possible. In the state of Illinois the percentage payback from slot machines in 2017 ranged from 89–92.5%, so in the long run you lose (thus the name "bandit"). But the slot machine paradigm provides a structure for thinking about optimal sequential decision making.

Multi-armed bandit (MAB) procedures address the problem of learning via experimentation which of k possible decisions leads to the greatest accumulated reward. Clearly there is a connection between R&S and MAB, and the procedures look similar, but they are not the same. The usual objective of MAB is to minimize "regret" (defined below) when making repeated decisions, while R&S attempts to identify the best system to implement. Most MAB procedures are intended for online use, while R&S is offline simulation optimization. MAB and R&S have different standards for "good procedure performance" and different assumptions about the reward/output data. R&S procedures tend to be more willing to waste observations on inferior systems so as to reduce the *overall* number of observations needed to make a correct selection, while MAB, which accumulates rewards, attempts to avoid the regret of choosing decisions with suboptimal rewards while searching for the best decision. A good overview reference is [16]. There is no denying that "multi-armed bandit" is a cooler name than "ranking & selection," but both have their roles.

In a bit more detail, "online" means making decisions in real time, with a stochastic reward after each decision, while "offline" means running a computer experiment to select a system and then implementing the selection in the real world. There is no reward associated with the R&S experiment, although there is a computational cost for running simulation.

The term "regret" refers to the shortfall in rewards that are obtained relative to what could have been attained by making the best decision, while PCS refers to getting the best choice in the end, not how one gets there. MAB tends to evaluate procedures via their probability *complexity*, while R&S evaluates procedures via their *finite-time effort*. MAB tends to assume sub-Gaussian (even bounded) reward distributions; R&S often assumes normally distributed output. MAB typically assumes a finite budget, and R&S often desires fixed precision.

In the classical stochastic MAB formulation, using our R&S notation, the decisions or "arms" are $x \in \{1, 2, \ldots, k\}$, with unknown reward distribution F_x having expected value $\mu(x)$ for making decision x. Let I_t be the decision chosen on opportunity t, and $Y_t(I_t) \sim F_{I_t}$ the associated stochastic reward. Using this notation, Table 2 defines regret, expected regret, and pseudo-regret. Loosely, the goal is to pick a policy for selecting I_t that minimizes regret or expected regret; since neither of these is achievable, pseudo-regret is a stand-in. One well-known MAB procedure is the upper confidence bound (UCB) policy: At the end of decision opportunity t, construct an

Table 2 MAB definitions of "regret."

Regret	$R_n = \max_x \sum_{t=1}^{n} Y_t(x) - \sum_{t=1}^{n} Y_t(I_t)$
Expected regret	$r_n = \mathrm{E}(R_n)$
Pseudo-regret	$\bar{r}_n = \max_x \mathrm{E}\left[\sum_{t=1}^{n} Y_t(x) - \sum_{t=1}^{n} Y_t(I_t)\right]$

UCB for each decision's mean, $\mu(x)$. On opportunity $t+1$, play the arm with the largest UCB. This is sometimes referred to as "optimism in the face of uncertainty" since one selects the decision with the largest apparent upside.

Clearly all forms of regret are non-decreasing in the number of decisions n that one makes; a good MAB procedure tries to have regret increase at the slowest possible rate. A building block result for pseudo-regret is the following:

$$\begin{aligned}\bar{r}_n &= n\mu(k) - \sum_{t=1}^{n} \mathrm{E}(\mu_{I_t}) \\ &= n\mu(k) - \sum_{x=1}^{k} \mu(x)\mathrm{E}(\#\text{ times played arm } x \text{ thru turn } n) \\ &= \sum_{x=1}^{k} (\mu(k) - \mu(x))\mathrm{E}(\#\text{ times played arm } x \text{ thru turn } n).\end{aligned}$$

One then derives an upper bound on the rate at which the $\mathrm{E}(\#\text{ times played arm } x \text{ thru turn } n)$ increases as the number of decisions n increases for $x \neq k$. This bounds the rate at which $\bar{r}_n$ increases. Note that this bound is neither an estimate of the pseudo-regret $\bar{r}_n$ nor a statistical guarantee. But it does say that as you play you accumulate regret no faster than the derived rate.

MAB procedures are frequently quite simple to implement, which makes them attractive, and of course many problems require online solutions (e.g., if you do not have the luxury of a simulation model of the world). However, for simulation optimization used to design systems, the R&S formulation tends to be more efficient and attacks the relevant objective.

Acknowledgements Special thanks to David Eckman and Linda Pei for comments on the structure and specifics of this tutorial, as well as to the two referees for their careful reports. This research was supported by National Science Foundation Grant Number DMS-1854562.

References

1. Bechhofer, R., Santner, T., Goldsman, D.: Design and Analysis of Experiments for Statistical Selection, Screening, and Multiple Comparisons. Wiley, New York (1995)
2. Chen, C.H., Chick, S.E., Lee, L.H., Pujowidianto, N.A.: Ranking and selection: efficient simulation budget allocation. In: Fu, M. (ed.) Handbook of Simulation Optimization, pp. 45–80. Springer (2015)
3. Chen, C.H., Lee, L.H.: Stochastic Simulation Optimization: An Optimal Computing Budget Allocation. World Scientific, Singapore (2011)
4. Chen, E.J.: Using parallel and distributed computing to increase the capability of selection procedures. In: Proceedings of the 2005 Winter Simulation Conference, pp. 723–731. IEEE (2005)
5. Chen, Y., Ryzhov, I.O.: Complete expected improvement converges to an optimal budget allocation. Adv. Appl. Probab. **51**(1), 209–235 (2019)
6. Dieker, A., Kim, S.H.: Selecting the best by comparing simulated systems in a group of three when variances are known and unequal. In: Proceedings of the 2012 Winter Simulation Conference, pp. 490–496. IEEE (2012)
7. Eckman, D.J., Henderson, S.G.: Guarantees on the probability of good selection. In: Proceedings of the 2018 Winter Simulation Conference, pp. 351–365. IEEE (2018)
8. Efron, B.: Large-scale Inference: Empirical Bayes Methods for Estimation, Testing, and Prediction. Cambridge University Press (2012)
9. Fan, W., Hong, L.J., Nelson, B.L.: Indifference-zone-free selection of the best. Oper. Res. **64**(6), 1499–1514 (2016)
10. Frazier, P.: Tutorial: Optimization via simulation with Bayesian statistics and dynamic programming. In: Proceedings of the 2012 Winter Simulation Conference, pp. 79–94. IEEE (2012)
11. Glynn, P., Juneja, S.: A large deviations perspective on ordinal optimization. In: Proceedings of the 2004 Winter Simulation Conference, pp. 577–585. IEEE (2004)
12. Goldsman, D.: Ranking and selection in simulation. In: Proceedings of the 1983 Winter Simulation Conference, pp. 387–393. IEEE (1983)
13. Gupta, S.S., Panchapakesan, S.: Multiple Decision Procedures: Theory and Methodology of Selecting and Ranking Populations. SIAM, Philadelphia (2002)
14. Hong, L.J.: Fully sequential indifference-zone selection procedures with variance-dependent sampling. Nav. Res. Logist. **53**(5), 464–476 (2006)
15. Hunter, S.R., Nelson, B.L.: Parallel ranking and selection. In: Advances in Modeling and Simulation, pp. 249–275. Springer, New York (2017)
16. Jamieson, K., Nowak, R.: Best-arm identification algorithms for multi-armed bandits in the fixed confidence setting. In: 48th Annual Conference on Information Sciences and Systems (CISS), pp. 1–6. IEEE (2014)
17. Jennison, C., Johnstone, I.M., Turnbull, B.W.: Asymptotically optimal procedures for sequential adaptive selection of the best of several normal means. In: Statistical Decision Theory and Related Topics III, pp. 55–86. Elsevier (1982)
18. Kamiński, B., Szufel, P.: On parallel policies for ranking and selection problems. J. Appl. Stat. **45**(9), 1690–1713 (2018)
19. Kim, S.H., Nelson, B.L.: A fully sequential procedure for indifference-zone selection in simulation. ACM Trans. Model. Comput. Simul. (TOMACS) **11**(3), 251–273 (2001)
20. Kim, S.H., Nelson, B.L.: On the asymptotic validity of fully sequential selection procedures for steady-state simulation. Oper. Res. **54**(3), 475–488 (2006)
21. Kim, S.H., Nelson, B.L.: Selecting the best system. In: Henderson, S.G., Nelson, B.L. (eds.) Handbooks in Operations Research and Management Science, vol. 13, pp. 501–534. Elsevier (2006)
22. L'Ecuyer, P., Simard, R., Chen, E.J., Kelton, W.D.: An object-oriented random-number package with many long streams and substreams. Oper. Res. **50**(6), 1073–1075 (2002)
23. Lee, S., Nelson, B.L.: General-purpose ranking and selection for computer simulation. IIE Trans. **48**(6), 555–564 (2016)

24. Luo, J., Hong, L.J., Nelson, B.L., Wu, Y.: Fully sequential procedures for large-scale ranking-and-selection problems in parallel computing environments. Oper. Res. **63**(5), 1177–1194 (2015)
25. Luo, Y.C., Chen, C.H., Yücesan, E., Lee, I.: Distributed web-based simulation optimization. In: Proceedings of the 2000 Winter Simulation Conference, pp. 1785–1793. IEEE (2000)
26. Mukhopadhyay, N., Solanky, T.K.: Multistage Selection and Ranking Procedures: Second Order Asymptotics. CRC Press, Boca Raton, Florida (1994)
27. Nelson, B.L.: Selecting the best simulated system: thinking differently about an old problem. In: International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing, pp. 69–79. Springer, New York (2018)
28. Nelson, B.L., Matejcik, F.J.: Using common random numbers for indifference-zone selection and multiple comparisons in simulation. Manage. Sci. **41**(12), 1935–1945 (1995)
29. Nelson, B.L., Pei, L.: Foundations and Methods of Stochastic Simulation: A First Course, 2nd edn. Springer, New York (2021)
30. Nelson, B.L., Swann, J., Goldsman, D., Song, W.: Simple procedures for selecting the best simulated system when the number of alternatives is large. Oper. Res. **49**(6), 950–963 (2001)
31. Ni, E.C., Ciocan, D.F., Henderson, S.G., Hunter, S.R.: Efficient ranking and selection in parallel computing environments. Oper. Res. **65**(3), 821–836 (2017)
32. Paulson, E.: A sequential procedure for selecting the population with the largest mean from k normal populations. Ann. Math. Stat. 174–180 (1964)
33. Pei, L., Hunter, S.R., Nelson, B.L.: A new framework for parallel ranking & selection using an adaptive standard. In: Proceedings of the 2018 Winter Simulation Conference, pp. 2201–2212. IEEE (2018)
34. Pei, L., Nelson, B.L., Hunter, S.R.: Evaluation of bi-PASS for parallel simulation optimization. In: Proceedings of the 2020 Winter Simulation Conference, pp. 2960–2971. IEEE (2020)
35. Rinott, Y.: On two-stage selection procedures and related probability-inequalities. Commun. Stat.-Theory Methods **7**(8), 799–811 (1978)
36. Salemi, P., Song, E., Nelson, B.L., Staum, J.: Gaussian Markov random fields for discrete optimization via simulation: framework and algorithms. Oper. Res. **67**(1), 250–266 (2019)
37. Zhong, Y., Hong, L.J.: Fully sequential ranking and selection procedures with PAC guarantee. In: Proceedings of the 2018 Winter Simulation Conference, pp. 1898–1908. IEEE (2018)
38. Zhong, Y., Hong, L.J.: Knockout-tournament procedures for large-scale ranking and selection in parallel computing environments. Oper. Res. **70**(1), 432–453 (2021)

Where are the Logs?

Art B. Owen and Zexin Pan

Abstract The commonly quoted error rates for QMC integration with an infinite low discrepancy sequence is $O(n^{-1}\log(n)^r)$ with $r = d$ for extensible sequences and $r = d - 1$ otherwise. Such rates hold uniformly over all d dimensional integrands of Hardy-Krause variation one when using n evaluation points. Implicit in those bounds is that for any sequence of QMC points, the integrand can be chosen to depend on n. In this paper we show that rates with any $r < (d-1)/2$ can hold when f is held fixed as $n \to \infty$. This is accomplished following a suggestion of Erich Novak to use some unpublished results of Trojan from the 1980s as given in the information based complexity monograph of Traub, Wasilkowski and Woźniakowski. The proof is made by applying a technique of Roth with the theorem of Trojan. The proof is non constructive and we do not know of any integrand of bounded variation in the sense of Hardy and Krause for which the QMC error exceeds $(\log n)^{1+\epsilon}/n$ for infinitely many n when using a digital sequence such as one of Sobol's. An empirical search when $d = 2$ for integrands designed to exploit known weaknesses in certain point sets showed no evidence that $r > 1$ is needed. An example with $d = 3$ and n up to 2^{100} might possibly require $r > 1$.

Keywords Discrepancy · Koksma-Hlawka inequality · Quasi-Monte Carlo · Trojan's theorem

1 Introduction

In this article, we study the asymptotic error rates for integration by quasi-Monte Carlo (QMC) as $n \to \infty$ while f is fixed. Most of the error upper bounds in QMC are based on fooling functions f_n that, given n integration points, are poorly integrated. By contrast, most of the published empirical results follow the integration error for a

A. B. Owen (✉) · Z. Pan
Stanford University, Stanford, CA, USA
e-mail: owen@stanford.edu

Z. Pan
e-mail: zep002@stanford.edu

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_19

single integrand f as n increases. The upper bounds have us play against an adaptive adversary choosing an unfavorable f_n at each sample size n instead of keeping f fixed as $n \to \infty$. The error bounds that we describe in more detail below are typically $O(\log(n)^r/n)$ where r can be as large as the dimension of the integrand's domain. These bounds can be enormous and, to our knowledge, there has never been an integrand exhibited where a standard QMC point set is shown to need $r > 1$. That raises the question of whether $r > 1$ is simply a consequence of the adversarial formulation. The alternative is that some function f is a 'persistent fooling function' causing large errors for infinitely many n. In an earlier version of this article we posed a question about whether *any* integrand of bounded variation in the sense of Hardy and Krause (BVHK) on $[0, 1]^d$ for any $d \geqslant 1$ has an integration error above $c\log(n)^r/n$ for infinitely many n with $r > 1$ and $c > 0$ using a digital sequence such as Sobol's for the integration points. For background on bounded variation in the sense of Hardy and Krause or in the sense of Vitali, we refer the reader to [18].

We owe a great debt to Erich Novak who pointed us to some unpublished work of Trojan described in detail in Chapter 10 of the information based complexity monograph of Traub, Wasilkowski and Woźniakowski [25]. Trojan's work is about very general problems of computing linear operators on Banach spaces based on the values of $n \to \infty$ linear functionals. He shows that the adversarial worst case convergence rate is also very nearly the attained rate for some specific problem instances. In the QMC context, that work pertains to a single integrand f as the number n of evaluation points diverges to infinity. A consequence of that work is that for any infinite sequence of integration points, there are indeed integrands in BVHK$[0, 1]^d$ with an absolute error larger than $c\log(n)^{(d-1)/2}/(n\log\log(n))$ infinitely often, for any $c > 0$. Furthermore, those integrands are present within a reproducing kernel Hilbert space (RKHS) on a certain unanchored space. They are dense in that space, though this does not mean that the usual Gaussian processes on such spaces give them positive measure. We only get $r = (d-1)/2$ logarithmic factors instead of d or $d-1$ of them. The explanation is that we use an L^2 bound just like Roth [21] used in getting a lower bound on star discrepancy. A different analysis might yield larger r. The $\log\log(n)$ factor in the denominator can be replaced by a sequence that diverges more slowly.

We have not been able to construct a function in BVHK$[0, 1]^d$ that provably needs $r > 1$ powers of $\log(n)$ for a Sobol' sequence [24] or the Halton [9] sequence, even when exploiting known weaknesses of commonly used QMC sequences. So, we are left to wonder: where are the logs?

An outline of this paper is as follows. Section 2 presents some results from the QMC literature and introduces notation on some QMC sequences. Section 3 proves our main result described above on existence of persistent fooling functions. Section 4 looks at the case $d = 1$ to exhibit some example functions requiring $r = 1$ for the van der Corput sequence: $f(x) = 1_{[0,2/3)}(x)$ and $f(x) = x$. Section 5 computes error for some $d = 2$ dimensional problems. The Halton and Sobol' points there are closely related to van der Corput points, yet two dimensional generalizations of the problematic integrands from Sect. 4 fail to show a need for $r > 1$. In fact some of the empirical results are more consistent with an $O(1/n)$ error. Section 6

computes a local discrepancy $\delta(\boldsymbol{z})$ for Sobol' nets with $d = 2, 3$ and $1 \leqslant m \leqslant 100$ where all components of $\boldsymbol{z}$ equal $2/3$ chosen because $2/3$ is difficult to approximate by dyadic rationals. It also includes $d = 4$ for $1 \leqslant m \leqslant 50$. The cases with $d > 2$ are the closest we have found to needing $r > 1$ but are inconclusive. Section 7 discusses these results.

Before we begin, we mention a note on our notation. The event that A occurs is commonly written as 1_A or $\mathbf{1}_A$ in works on probability. Because our expressions for A will include subscripts we make use a nonstandard $\mathbf{1}\{A\}$. For instance

$$\mathbf{1}\{x_{ij} < y_i\} = \begin{cases} 1, & x_{ij} < y_i \\ 0, & \text{else.} \end{cases}$$

We find this clearer than the Iverson notation $[x_{ij} < y_i]$ and more readable than having the event description within a subscript.

2 Background

From the Koksma-Hlawka inequality [10] combined with convergence rates for the star discrepancy [15], we get the widely quoted convergence rates for the error in quasi-Monte Carlo integration of a function $f : [0, 1]^d \to \mathbb{R}$. An integrand f of bounded variation in the sense of Hardy and Krause, written $f \in \mathrm{BVHK}[0, 1]^d$, can be integrated with error $O(n^{-1}(\log n)^{d-1})$ using n function evaluations. If we must use the first n points of an infinite sequence, then the rate $O(n^{-1}(\log n)^d)$ is attainable. This article is mostly about the infinite sequence version. Both of these rates are often written $O(n^{-1+\epsilon})$ where ϵ can be any positive constant but $\log(n)^d \gg n^\epsilon$ for many use cases of interest.

For high dimensional problems, such powers of $\log(n)$ are enormous and then there is genuine uncertainty about whether $O(n^{-1}(\log n)^d)$ is better than the root mean squared error (RMSE) of $O(n^{-1/2})$ from plain Monte Carlo (MC) at practically relevant n. These rates omit three implied constants: one in the star discrepancy (see [7] for information), one in the total variation of f and the third one is the standard deviation of f. These unknown constants contribute to uncertainty about the n at which QMC would outperform MC. A further complication is that the Koksma-Hlawka bound is for a worst case integrand. The situation is quite different in Monte Carlo (MC) where the rate $\sigma n^{-1/2}$ holds for all finite n making it simultaneously a guide to how accuracy progresses for a single integrand of variance σ^2 and the RMSE formula (upper and lower bound) for all integrands of variance σ^2.

That observed errors for realistic n and large d do not follow a trend like $\log(n)^d/n$ was reported by Schlier [22] among others. That work also found that the variance of $f(\boldsymbol{x})$ was more useful than its total variation in explaining the empirical accuracy of QMC integration on test functions, despite the fact that proved theoretical bounds for QMC error use total variation and variance does not require any of the smoothness

that QMC relies on. Many papers include empirically estimated convergence rates for individual f found by fitting a regression model for log error versus $\log(n)$. See for instance L'Ecuyer [12]. We do not see results that look like a large power of $\log(n)$ is present.

This mismatch between empirical results and theoretical ones is troubling. Empirical results alone don't give enough confidence that they will apply to future problems. Similarly, bounds that are favorable (but asymptotic) or unfavorable (but worst case) could also fail to provide a reliable guide to attained accuracy. This mismatch has brought practical difficulties. For instance, the logarithmic powers in the Koksma-Hlawka bound led Bratley, Fox and Niederreiter [1] to limit their software to $d \leqslant 12$.

For some randomizations of digital nets the RMSE is $O(n^{-1/2})$ whenever $f \in L^2[0,1]^d$ [17] and is also $O(\log(n)^{(d-1)/2}/n)$ under further smoothness conditions [16, 19, 28]. In such cases the large powers of $\log(n)$ are subject to a simultaneous $O(n^{-1/2})$ bound that limits how much worse randomized QMC can be compared to MC for finite n. It would be interesting to know whether something like that also holds for plain QMC. Perhaps the coefficient of $\log(n)^r/n$ is ordinarily very small, or the effect is only relevant for impractically large n or perhaps not even present for most commonly investigated integrands. For a survey of randomized QMC see L'Ecuyer and Lemieux [13].

We conclude this section by describing (t, m, d)-nets and (t, d)-sequences using the formulation from Niederreiter [14]. Let $b \geqslant 2$ be an integer. For $\boldsymbol{k} = (k_1, \ldots, k_d) \in \mathbb{N}_0^d$ and $\boldsymbol{c} = (c_1, \ldots, c_d) \in \mathbb{Z}_0^d$ with $0 \leqslant c_j < b^{k_j}$ the half open hyperrectangle

$$E(\boldsymbol{k}, \boldsymbol{c}) = \prod_{j=1}^{d} \left[\frac{c_j}{b^{k_j}}, \frac{c_j + 1}{b^{k_j}} \right) \tag{1}$$

is called an elementary interval in base b. It has volume $b^{-|\boldsymbol{k}|}$ where $|\boldsymbol{k}| = \sum_{j=1}^{d} k_j$. We define its indicator function as

$$I_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{x}) = I_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{x}; b) = \mathbf{1}\{\boldsymbol{x} \in E(\boldsymbol{k}, \boldsymbol{c})\}.$$

For integers $m \geqslant t \geqslant 0$, $b \geqslant 2$, $n = b^m$ and $d \geqslant 1$ the points $\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{n-1} \in [0,1]^d$ are a (t, m, d)-net in base b if

$$\frac{1}{n} \sum_{i=0}^{n-1} I_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{x}_i) = \int_{[0,1]^d} I_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = b^{-|\boldsymbol{k}|}$$

holds for all elementary intervals with $|\boldsymbol{k}| \leqslant m - t$. Other things being equal, smaller t are better and $t = 0$ is best, but the choices of d, b, and m impose a lower bound on the possible t which may rule out $t = 0$. The minT project of [23] tracks the best known values of t as well as some lower bounds on t.

In this paper we emphasize infinite sequences. The sequence $\boldsymbol{x}_i \in [0,1]^d$ for integers $i \geqslant 0$ is a (t,d)-sequence in base b if $\boldsymbol{x}_{rb^m}, \ldots, \boldsymbol{x}_{(r+1)b^m-1}$ is a (t,m,d)-net in base b for all $m \geqslant t$ and all integers $r \geqslant 0$. These are extensible (t,m,d)-nets in that the first b^ℓ points of a (t,d)-sequence form a $(t,m+\ell,d)$-net for any integer $\ell \geqslant 1$. The most used (t,d)-sequences are the $(0,d)$-sequences in prime bases $p \geqslant d$ of Faure [6] and the (t,d)-sequences in base 2 of Sobol' [24].

We will make special use of the van der Corput sequences in base $b \geqslant 2$. These are $(0,1)$-sequences in base b. If we write the natural number $i = \sum_{k=1}^{\infty} i_k b^{k-1}$ with digits $i_k \in \{0,1,\ldots,b-1\}$, then the sum only has $K(i) < \infty$ nonzero terms and we then set $x_i = \sum_{k=1}^{K(i)} b^{-k} i_k$. This sequence has star discrepancy $D_n^* = O(\log(n)/n)$. For $n = b^m$ it is a left endpoint rule containing points i/n for $0 \leqslant i < n$ and so it has $D_n^* = 1/n$ by [15, Theorem 2.6]. The original van der Corput sequence in base $b = 2$ is from [26].

3 Proof of the Lower Bound

We begin with a general theorem on worst-case errors. Later we specialize it to the QMC setting.

Theorem 1 *Let $(F, \|\cdot\|)$ be a Banach space and S be a linear functional on F. For a sequence of continuous linear functionals L_n on F, define*

$$N_n(f) = (L_1(f), \ldots, L_n(f)), \quad \textit{and}$$
$$r_n = \sup\bigl\{|S(f)| \;\big|\; f \in F, N_n(f) = \boldsymbol{0}, \|f\| \leqslant 1\bigr\}.$$

Then for any sequence of mappings ϕ_n from $\mathbb{R}^n$ to $\mathbb{R}$, there exists $f \in F$ such that

$$\limsup_{n\to\infty} \frac{|S(f) - \phi_n(N_n(f))|}{(\log\log n)^{-1} r_n} = +\infty. \tag{2}$$

Proof This follows from Theorem 2.1.1 in Chap. 10 of [25] who cite unpublished work by Trojan. □

We will set F to be some reproducing kernel Hilbert space (RKHS) contained in BVHK$[0,1]^d$, $S(f)$ to be $\int_{[0,1]^d} f(\boldsymbol{y})\,\mathrm{d}\boldsymbol{y}$ and $N_n(f)$ to be $(f(\boldsymbol{x}_0), \ldots, f(\boldsymbol{x}_{n-1}))$. We note that evaluation at $\boldsymbol{x}_i$ is a continuous linear functional in an RKHS. As the theorem suggests, we can use $r_n/\log\log n$ as a lower bound on the asymptotic convergence rate achievable by all functions in BVHK$[0,1]^d$, so it remains to determine a lower bound on r_n.

To derive such a lower bound, we will apply the proof techniques used in Roth's lower bound on the L^2 discrepancy. See Chen and Travaglini [2] for a nice summary. Dick, Hinrichs and Pillichshammer [4] use this strategy to prove that the worst-case error of any equal-weight quadrature rule is lower bounded by $\Omega(n^{-1}(\log n)^{(d-1)/2})$

when F is the RKHS with kernel $K(\boldsymbol{x}, \boldsymbol{y}) = \prod_{j=1}^{d}(1 + \min(x_j, y_j))$. Wozniakowski [27] points out that the same strategy still works if the equal weight requirement is removed. Below we illustrate Roth's technique by showing that $r_n = \Omega(n^{-1}(\log n)^{(d-1)/2})$ if F is chosen to be the RKHS with kernel

$$K(\boldsymbol{x}, \boldsymbol{y}) = \prod_{j=1}^{d}\Big(\frac{4}{3} + \frac{1}{2}\Big(x_j^2 + y_j^2 - x_j - y_j - |x_j - y_j|\Big)\Big). \tag{3}$$

This is the unanchored space introduced in [3]. It has the inner product

$$\begin{aligned}(f, g) &\tag{4}\\ &= \sum_{u\subseteq\{1,\dots,d\}} \int_{[0,1]^{|u|}} \bigg(\int_{[0,1]^{d-|u|}} \frac{\partial^{|u|} f}{\partial \boldsymbol{y}_u}(\boldsymbol{y})\,\mathrm{d}\boldsymbol{y}_{-u}\bigg)\bigg(\int_{[0,1]^{d-|u|}} \frac{\partial^{|u|} g}{\partial \boldsymbol{y}_u}(\boldsymbol{y})\,\mathrm{d}\boldsymbol{y}_{-u}\bigg)\,\mathrm{d}\boldsymbol{y}_u\end{aligned}$$

where $\partial^{|u|} f/\partial \boldsymbol{y}_u$ is the partial derivative of f taken once with respect to each y_j with $j \in u$. Any f belonging to this RKHS has mixed partial derivative $\partial^{|u|} f/\partial \boldsymbol{y}_u \in L^2$ for any $u \subseteq \{1, \dots, d\}$, so f belongs to BVHK$[0, 1]^d$ using Eq. (5) and Proposition 13 of [18].

Letting $\mathbf{1} \in F$ be the function equal to 1 for all $\boldsymbol{y} \in [0, 1]^d$, it is straightforward to verify that

$$(f, \mathbf{1}) = \int_{[0,1]^d} f(\boldsymbol{y})\,\mathrm{d}\boldsymbol{y} = S(f).$$

In other words, the function $\mathbf{1}$ is the Riesz representation of integration over $[0, 1]^d$. Moreover, by the reproducing property $(f, K(\boldsymbol{x}_i, \cdot)) = f(\boldsymbol{x}_i)$. Therefore

$$\begin{aligned} r_n &= \sup\big\{|S(f)| \mid f \in F, f(\boldsymbol{x}_0) = \cdots = f(\boldsymbol{x}_{n-1}) = 0, \|f\| \leqslant 1\big\}\\ &= \sup\big\{|(f, \mathbf{1})| \mid f \in F, (f, K(\boldsymbol{x}_0, \cdot)) = \cdots = (f, K(\boldsymbol{x}_{n-1}, \cdot)) = 0, \|f\| \leqslant 1\big\}.\end{aligned}$$

This is the well-known least squares projection problem. The maximizer is proportional to the projection of $\mathbf{1}$ into the orthogonal complement of the linear span of $\{K(\boldsymbol{x}_0, \cdot), \dots, K(\boldsymbol{x}_{n-1}, \cdot)\}$. Therefore

$$r_n = \min_{a_0,\dots,a_{n-1}} \Big\|\mathbf{1} - \sum_{i=0}^{n-1} a_i K(\boldsymbol{x}_i, \cdot)\Big\|.$$

Now we prove that $\|\mathbf{1} - \sum_{i=0}^{n-1} a_i K(\boldsymbol{x}_i, \cdot)\| = \Omega(n^{-1}(\log n)^{(d-1)/2})$ for any choice of $a_0, \dots, a_{n-1}$, including $a_i = 1/n$ as used in QMC.

Theorem 2 *Let K be the kernel (3) for the unanchored RKHS. For any points $\boldsymbol{x}_0, \dots, \boldsymbol{x}_{n-1} \in [0, 1]^d$ and any weights $a_0, \dots, a_{n-1} \in \mathbb{R}$*

$$\left\| 1 - \sum_{i=0}^{n-1} a_i K(\boldsymbol{x}_i, \cdot) \right\| \geqslant A_d \frac{(\log n)^{(d-1)/2}}{n} \tag{5}$$

holds for some positive number A_d independent of n.

Proof Because the function $\mathbf{1}$ is the Riesz representation of integration over $[0, 1]^d$, $(\mathbf{1}, \mathbf{1}) = \int_{[0,1]^d} \mathbf{1}\, \mathrm{d}\boldsymbol{y} = 1$, and

$$(K(\boldsymbol{x}, \cdot), 1) = \prod_{j=1}^{d} \int_0^1 \frac{4}{3} + \frac{1}{2}\Big(x_j^2 + y_j^2 - x_j - y_j - |x_j - y_j|\Big)\, \mathrm{d}y_j = 1.$$

Therefore

$$\Big(\mathbf{1} - \sum_{i=0}^{n-1} a_i K(\boldsymbol{x}_i, \cdot), \mathbf{1}\Big) = (\mathbf{1}, \mathbf{1}) - \sum_{i=0}^{n-1} a_i (K(\boldsymbol{x}_i, \cdot), \mathbf{1}) = 1 - \sum_{i=0}^{n-1} a_i.$$

On the other hand,

$$\left|\Big(\mathbf{1} - \sum_{i=0}^{n-1} a_i K(\boldsymbol{x}_i, \cdot), \mathbf{1}\Big)\right| \leqslant \|\mathbf{1}\| \times \left\| \mathbf{1} - \sum_{i=0}^{n-1} a_i K(\boldsymbol{x}_i, \cdot) \right\| = \left\| \mathbf{1} - \sum_{i=0}^{n-1} a_i K(\boldsymbol{x}_i, \cdot) \right\|,$$

so we get the first lower bound

$$\left\| 1 - \sum_{i=0}^{n-1} a_i K(\boldsymbol{x}_i, \cdot) \right\| \geqslant \left| 1 - \sum_{i=0}^{n-1} a_i \right|. \tag{6}$$

Next we evaluate $\|1 - \sum_{i=0}^{n-1} a_i K(\boldsymbol{x}_i, \cdot)\|^2$ directly. If we ignore all summands except for $u = \{1, \ldots, d\}$ in (4), we get

$$\begin{aligned} \left\| 1 - \sum_{i=0}^{n-1} a_i K(\boldsymbol{x}_i, \cdot) \right\|^2 &\geqslant \int_{[0,1]^d} \left(\sum_{i=0}^{n-1} a_i \frac{\partial^d K(\boldsymbol{x}_i, \boldsymbol{y})}{\partial y_1 \cdots \partial y_d} \right)^2 \mathrm{d}\boldsymbol{y} \\ &= \int_{[0,1]^d} \left(\sum_{i=0}^{n-1} a_i \prod_{j=1}^{d} (y_j - \mathbf{1}\{y_j > x_{ij}\}) \right)^2 \mathrm{d}\boldsymbol{y} \end{aligned} \tag{7}$$

where x_{ij} is the jth component of $\boldsymbol{x}_i$.

The left hand side of (7) is a squared norm in the RKHS while the right hand side is a plain L^2 squared norm. To provide a lower bound, we construct a function h satisfying

$$\int_{[0,1]^d} h(\boldsymbol{y})^2 \,\mathrm{d}\boldsymbol{y} = O((\log n)^{d-1}), \quad \text{and}$$
$$\int_{[0,1]^d} h(\boldsymbol{y})\Big(\sum_{i=0}^{n-1} a_i \prod_{j=1}^{d}(y_j - \mathbf{1}\{y_j > x_{ij}\})\Big)\,\mathrm{d}\boldsymbol{y} = \Omega\Big(\frac{(\log n)^{d-1}}{n}\Big),$$

neither of which involve the RKHS inner product, so the function h does not have to be in the RKHS.

Define $E(\boldsymbol{k}, \boldsymbol{c})$ to be the d-dimensional interval from (1) with $b = 2$, that is

$$E(\boldsymbol{k}, \boldsymbol{c}) = \prod_{j=1}^{d}\Big[\frac{c_j}{2^{k_j}}, \frac{c_j+1}{2^{k_j}}\Big)$$

where $\boldsymbol{k} = (k_1, \ldots, k_d) \in \mathbb{Z}^d$ and $\boldsymbol{c} = (c_1, \ldots, c_d) \in \mathbb{Z}^d$ satisfy $k_j \geqslant 0$ and $0 \leqslant c_j < 2^{k_j}$. Given $\boldsymbol{k}$, we define $|\boldsymbol{k}| = \sum_{j=1}^{d} k_j$. For a given vector $\boldsymbol{k}$, the $2^{|\boldsymbol{k}|}$ elementary intervals $E(\boldsymbol{k}, \boldsymbol{c})$ partition $[0, 1)^d$ into congruent sub-intervals.

For each $E(\boldsymbol{k}, \boldsymbol{c})$, define $U_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{y})$ by

$$U_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{y}) = \begin{cases} (-1)^{\sum_{j=1}^{d} \mathbf{1}\{2^{k_j} y_j - c_j < 1/2\}}, & \boldsymbol{y} \in E(\boldsymbol{k}, \boldsymbol{c}) \\ 0, & \text{else.} \end{cases}$$

If we partition $E(\boldsymbol{k}, \boldsymbol{c})$ into 2^d congruent sub-intervals, by splitting each side $[c_j/2^{k_j}, (c_j+1)/2^{k_j})$ at its midpoint, then the value of $U_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{y})$ is constant on each such sub-interval. Sub-intervals of $E(\boldsymbol{k}, \boldsymbol{c})$ that share a $d-1$ dimensional face have the opposite sign for $U_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{y})$.

It is straightforward to verify that $\int_{[0,1]^d} U_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{y}) U_{\boldsymbol{k}',\boldsymbol{c}'}(\boldsymbol{y})\,\mathrm{d}\boldsymbol{y} = 0$ if $\boldsymbol{k} \neq \boldsymbol{k}'$ or $\boldsymbol{c} \neq \boldsymbol{c}'$. It is trivially true if $E(\boldsymbol{k}, \boldsymbol{c}) \cap E(\boldsymbol{k}', \boldsymbol{c}') = \varnothing$. If instead $E(\boldsymbol{k}, \boldsymbol{c}) \cap E(\boldsymbol{k}', \boldsymbol{c}') \neq \varnothing$, then there must be some j with $k_j < k_j'$ and observe that as a function of y_j, $U_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{y}) U_{\boldsymbol{k}',\boldsymbol{c}'}(\boldsymbol{y})$ equals 1 on a $1/2^{k_j+1}$-length interval and equals -1 on an adjacent $1/2^{k_j+1}$-length interval, so the integration over variable j always returns 0. Then for any set P of $(\boldsymbol{k}, \boldsymbol{c})$ pairs,

$$\int_{[0,1]^d}\Big(\sum_{(\boldsymbol{k},\boldsymbol{c})\in P} U_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{y})\Big)^2 \mathrm{d}\boldsymbol{y} = \sum_{(\boldsymbol{k},\boldsymbol{c})\in P}\int_{[0,1]^d} U_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{y})^2\,\mathrm{d}\boldsymbol{y} = \sum_{(\boldsymbol{k},\boldsymbol{c})\in P} 2^{-|\boldsymbol{k}|}. \tag{8}$$

Now let $\mathcal{P} = \{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{n-1}\}$ and choose m so that $2n \leqslant 2^m < 4n$. For any $\boldsymbol{k}$ with $|\boldsymbol{k}| = m$, define the set $P_{\boldsymbol{k}}$ to be

$$P_{\boldsymbol{k}} = \{\boldsymbol{c} \mid E(\boldsymbol{k}, \boldsymbol{c}) \cap \mathcal{P} = \varnothing\}.$$

For each $\boldsymbol{c} \in P_{\boldsymbol{k}}$,

$$\int_{[0,1]^d} U_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{y}) \prod_{j=1}^{d} (y_j - \mathbf{1}\{y_j > x_{ij}\})\, \mathrm{d}\boldsymbol{y} = \frac{1}{4^{m+d}}.$$

Because there are 2^m intervals associated with $\boldsymbol{k}$, the cardinality of $P_{\boldsymbol{k}}$ is at least $2^m - n \geqslant n$. Hence

$$\int_{[0,1]^d} \left(\sum_{\boldsymbol{c} \in P_{\boldsymbol{k}}} U_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{y}) \right) \prod_{j=1}^{d} (y_j - \mathbf{1}\{y_j > x_{ij}\})\, \mathrm{d}\boldsymbol{y} \geqslant \frac{n}{4^{m+d}}. \tag{9}$$

Now we define

$$h(\boldsymbol{y}) = \sum_{\boldsymbol{k}:|\boldsymbol{k}|=m} \sum_{\boldsymbol{c} \in P_{\boldsymbol{k}}} U_{\boldsymbol{k},\boldsymbol{c}}(\boldsymbol{y}).$$

The number of $\boldsymbol{k}$ with $|\boldsymbol{k}| = m$ is the number of ways to partition m into d nonnegative ordered integers, which equals $\binom{m+d-1}{d-1}$. Equation (8) and $2n \leqslant 2^m < 4n$ imply that

$$\int_{[0,1]^d} h(\boldsymbol{y})^2\, \mathrm{d}\boldsymbol{y} = \sum_{\boldsymbol{k}:|\boldsymbol{k}|=m} \sum_{\boldsymbol{c} \in P_{\boldsymbol{k}}} 2^{-m} \leqslant \sum_{\boldsymbol{k}:|\boldsymbol{k}|=m} 1 = \binom{m+d-1}{d-1} \leqslant C_d (\log n)^{d-1}$$

for some positive number C_d independent of n. On the other hand, Eq. (9) implies that

$$\int_{[0,1]^d} h(\boldsymbol{y}) \prod_{j=1}^{d} (y_j - \mathbf{1}\{y_j > x_{ij}\})\, \mathrm{d}\boldsymbol{y} \geqslant \binom{m+d-1}{d-1} \frac{n}{4^{m+d}} \geqslant \frac{c_d (\log n)^{d-1}}{n}$$

for another positive number c_d independent of n.

By the Cauchy-Schwarz inequality and Eq. (7)

$$\begin{aligned}
&\int_{[0,1]^d} h(\boldsymbol{y}) \left(\sum_{i=0}^{n-1} a_i \prod_{j=1}^{d} (y_j - \mathbf{1}\{y_j > x_{ij}\}) \right) \mathrm{d}\boldsymbol{y} \\
&\leqslant \left(\int_{[0,1]^d} h(\boldsymbol{y})^2\, \mathrm{d}\boldsymbol{y} \right)^{\frac{1}{2}} \left(\int_{[0,1]^d} \left(\sum_{i=0}^{n-1} a_i \prod_{j=1}^{d} (y_j - \mathbf{1}\{y_j > x_{ij}\}) \right)^2 \mathrm{d}\boldsymbol{y} \right)^{\frac{1}{2}} \\
&\leqslant \left(\int_{[0,1]^d} h(\boldsymbol{y})^2\, \mathrm{d}\boldsymbol{y} \right)^{\frac{1}{2}} \left\| \mathbf{1} - \sum_{i=0}^{n-1} a_i K(\boldsymbol{x}_i, \cdot) \right\|
\end{aligned}$$

which provides the lower bound

$$\left\|1-\sum_{i=0}^{n-1}a_iK(\boldsymbol{x}_i,\cdot)\right\| \geqslant (C_d(\log n)^{d-1})^{-\frac{1}{2}}\frac{c_d(\log n)^{d-1}}{n}\sum_{i=0}^{n-1}a_i$$
$$=\left(\frac{c_d}{C_d^{1/2}}\sum_{i=0}^{n-1}a_i\right)\frac{(\log n)^{(d-1)/2}}{n}.$$

Combining the above lower bound with Eq. (6) we get

$$\left\|\mathbf{1}-\sum_{i=0}^{n-1}a_iK(\boldsymbol{x}_i,\cdot)\right\| \geqslant \max\left(\left|1-\sum_{i=0}^{n-1}a_i\right|,\left(\frac{c_d}{C_d^{1/2}}\sum_{i=0}^{n-1}a_i\right)\frac{(\log n)^{(d-1)/2}}{n}\right).$$

For $\lambda > 0$,

$$\min_{a\in\mathbb{R}}\max(|1-a|,\lambda a)=\frac{\lambda}{\lambda+1}$$

so that

$$\left\|1-\sum_{i=0}^{n-1}a_iK(\boldsymbol{x}_i,\cdot)\right\| \geqslant \frac{c_d(\log n)^{(d-1)/2}/(nC_d^{1/2})}{1+c_d(\log n)^{(d-1)/2}/(nC_d^{1/2})}$$

and we let $A_d = (c_d/C_d^{1/2})/(1+c_dM_d/C_d^{1/2})$ for $M_d = \sup_{n\in\mathbb{N}}\log(n)^{(d-1)/2}/n$. □

Corollary 1 *For any sequence of points $(\boldsymbol{x}_i)_{i\geqslant 0}$ in $[0,1]^d$, there exists a function f in the RKHS with kernel defined by (3) such that*

$$\limsup_{n\to\infty}\frac{\left|\int_{[0,1]^d}f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}-\frac{1}{n}\sum_{i=0}^{n-1}f(\boldsymbol{x}_i)\right|}{(n\log\log n)^{-1}(\log n)^{(d-1)/2}}=+\infty$$

Proof Apply Theorem 1 with the lower bound on r_n from Theorem 2. □

4 Discrepancy and the Case of $d = 1$

Let $x_0, x_1, \ldots, x_{n-1} \in [0,1]$. The local discrepancy of these points at $\alpha \in [0,1]$ is

$$\delta_n(\alpha)=\frac{1}{n}\sum_{i=0}^{n-1}\mathbf{1}\{x_i<\alpha\}-\alpha$$

and the star discrepancy is $D_n^* = \sup_{0\leqslant\alpha\leqslant 1}|\delta_n(\alpha)|$. No infinite sequence x_i can have $D_n^* = o(\log(n)/n)$. Using results from discrepancy theory we see below that there are specific values of α for which $D_n^* = \Omega(\log(n)/n)$. For those values $\mathbf{1}\{x<\alpha\}-\alpha$

is a persistent fooling function. We show below that $f(x) = x$ is also a persistent fooling function for the van der Corput sequence.

The set of $\alpha \in [0, 1]$ with $|\delta_n(\alpha)| = o(\log(n)/n)$ has Hausdorff dimension 0 for any sequence $(x_i)_{i\geqslant 0} \subset [0, 1]$. See [8]. So $r = 1$ is not just available for $d = 1$ it is the usual rate for functions of the form $f(x) = \mathbf{1}\{x < \alpha\}$.

For x_i taken from the van der Corput sequence, and $\alpha = \sum_{k=1}^{\infty} a_k/2^k$ for bits $a_k \in \{0, 1\}$, Drmota, Larcher and Pillichshammer [5] note that $n|\delta_n(\alpha)|$ is bounded as $n \to \infty$, if and only if α has a representation with only finitely many nonzero a_k. Further, letting

$$h_\alpha(m) = \#\{k < m \mid a_k \neq a_{k+1}\}$$

their Corollary 1 in our notation has

$$\lim_{m\to\infty} \frac{1}{2^m} \#\left\{1 \leqslant n \leqslant 2^m \mid n\delta_n > (1-\epsilon)h_\alpha(m)\right\} = 1 \tag{10}$$

for any $\epsilon > 0$. The base 2 representation of $2/3$ is $0.10101\cdots$ and so $h_{2/3}(m) = m$. It follows that $f(x) = \mathbf{1}\{x < 2/3\}$ has $|\hat{\mu}_n - \mu| > c\log(n)/n$ infinitely often for some $c > 0$. Even more, the fraction of such n among the first $N = 2^m$ sample sizes becomes ever closer to 1 as $m \to \infty$.

If we average the local discrepancy over α we get

$$\int_0^1 \delta_n(\alpha)\,\mathrm{d}\alpha = \frac{1}{n}\sum_{i=0}^{n} x_i - \frac{1}{2}$$

which is the integration error for the function $f(x) = x$ that we study next. In our study of $f(x) = x$, we use sample sizes n with base 2 expansion $10101\cdots101$. That is, for some $L \geqslant 1$

$$n = n_L = \sum_{\ell=0}^{L} 4^\ell.$$

The first few values of n_L are 1, 5, 21, 85, 341, 1365, and 5461.

Proposition 1 *For integers $0 \leqslant i < n$, let x_i be the i'th van der Corput point in base 2. Then*

$$\sum_{i=0}^{n-1} x_i = \sum_{k=1}^{\infty} \lfloor 2^{-k-1}n \rfloor + (2^{-k}n - 2\lfloor 2^{-k-1}n \rfloor - 1)_+. \tag{11}$$

Proof As i increases from 0, the k'th digit of x_i comes in alternating blocks of 2^k zeros and 2^k ones, starting with zeros. For $0 \leqslant i < r2^{k+1}$ with an integer $r \geqslant 0$, the k'th digits sum to $r2^k$ because there are r complete blocks of 2^k ones. The number of ones among i_k for $0 \leqslant i < n$ is then

$$2^k\lfloor 2^{-k-1}n\rfloor + (n - 2^{k+1}\lfloor 2^{-k-1}n\rfloor - 2^k)_+$$

where $z_+ = \max(z, 0)$. The first term counts ones from $\lfloor 2^{-k-1}n\rfloor$ complete blocks of 2^{k+1} indices. That leaves an incomplete block of $n - 2^{k+1}\lfloor 2^{-k-1}n\rfloor$ indices i of which the first 2^k, should there be that many, must be 0s. Any indices past the first 2^k are ones providing the second term above. To complete the proof, the k'th digits have a coefficient of 2^{-k} in x_i and summing over digits yields (11). □

The sum over k in (11) only needs to go as far as the number of nonzero binary digits in $n - 1$. For larger k, both parts of the k'th term are zero.

Proposition 2 *For $L \geqslant 0$, let $n_L = \sum_{\ell=0}^{L} 4^\ell$. For integers $i \geqslant 0$ let x_i be the van der Corput points in base 2 and for $n \geqslant 1$ let $\hat{\mu}_n = (1/n)\sum_{i=0}^{n-1} x_i$. Then*

$$\limsup_{L\to\infty} \frac{n_L}{\log(n_L)} |\hat{\mu}_{n_L} - 1/2| > c$$

if $0 < c < 1/(8\log(2))$.

Proof We need $K(n_L) = 2L - 1$ base 2 digits to represent n_L. We can write $n_L = \sum_{\ell=0}^{L} 2^{2\ell}$. Then for $1 \leqslant k \leqslant 2L - 1$,

$$\lfloor 2^{-k-1}n_L\rfloor = \Big\lfloor 2^{-k-1}\sum_{\ell=0}^{L} 2^{2\ell}\Big\rfloor = 2^{-k-1}n_L - \sum_{\ell=0}^{\lfloor (k-1)/2\rfloor} 2^{2\ell-k-1}.$$

Next let

$$\theta_k \equiv \sum_{\ell=0}^{\lfloor (k-1)/2\rfloor} 2^{2\ell-k-1} = 2^{-k-1}\sum_{\ell=0}^{\lfloor (k-1)/2\rfloor} 2^{2\ell} = 2^{-k-1}\frac{4^{\lfloor (k+1)/2\rfloor} - 1}{3}$$

and then

$$(2^{-k}n_L - 2\lfloor 2^{-k-1}n_L\rfloor - 1)_+ = (2^{-k}n_L - 2^{-k}n_L + 2\theta_k - 1)_+ = (2\theta_k - 1)_+$$

Because $\theta_k < 1/3$ we have $(2\theta_k - 1)_+ = 0$. Using Proposition 1,

$$\begin{aligned}\frac{1}{n_L}\sum_{i=0}^{n_L-1} x_i &= \frac{1}{n_L}\sum_{k=1}^{K(n_L)} \lfloor 2^{-k-1}n_L\rfloor = \frac{1}{n_L}\sum_{k=1}^{K(n_L)} 2^{-k-1}n_L - \theta_k \\ &= \frac{1}{2} - 2^{-K(n_L)-1} - \frac{1}{n_L}\sum_{k=1}^{K(n_L)} \theta_k.\end{aligned}$$

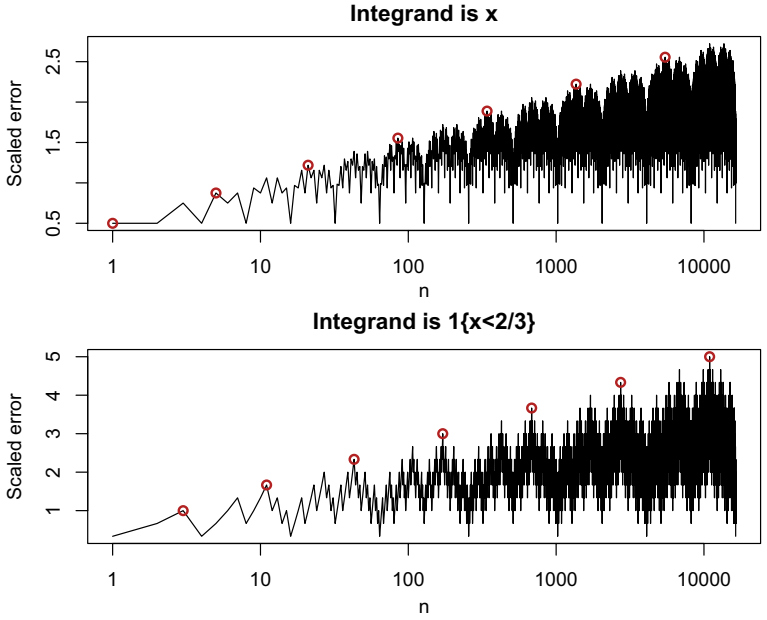


Fig. 1 Both panels show the scaled error $n|\hat{\mu}_n - \mu|$ versus n for the first 2^{14} points of the van der Corput sequence. The top panel has the integrand $f(x) = x$ and the values for $n = n_L$ are indicated with open circles. The bottom panel has the integrand $f(x) = \mathbf{1}\{x < 2/3\}$ and the values $n = \tilde{n}_L$ are indicated

Now because $\theta_k \geqslant \theta_2 = 1/8$, $|\hat{\mu}_{n_L} - 1/2| \geqslant K(n_L)/(8n_L)$ and so

$$\frac{n_L}{\log(n_L)}|\hat{\mu}_{n_L} - 1/2| \geqslant \frac{K(n_L)}{8\log(n_L)} > \frac{1}{8}\frac{2L-1}{(L+1)\log(4)} \to \frac{1}{4\log(4)},$$

completing the proof. □

We can see why the sample sizes n_L give unusually inaccurate estimates of the integral of x in the van der Corput sequence. Those values of n consistently miss getting into the 'ones block' for digit k.

Figure 1 shows some empirical behavior of the scaled errors for the two integrands we considered in this section. The scaled error there is essentially the number of observations by which the count of points in $[0, 2/3)$ differs from $2n/3$. It is $n|\delta_n(2/3)|$ which is the customary scaling in the discrepancy literature.

The integrands x and $\mathbf{1}\{x < 2/3\}$ both have total variation one. It would be interesting to know what integrand of total variation one has the largest value for $\limsup_{n\to\infty} n|\hat{\mu}_n - \mu|/\log(n)$ in the van der Corput sequence.

For integration, it is advisable to use $n = b^m$ for $m \geqslant 0$ in the van der Corput sequence. Then the Koksma-Hlawka inequality gives us $|\hat{\mu}_{b^m} - \mu| = O(1/b^m)$ as $m \to \infty$ for any f of bounded variation on $[0, 1]$ because the van der Corput sequence

has $D^*_{b^m} = b^{-m} = 1/n$. Any (t, d)-sequence for $d = 1$ has $D^*_{b^m} = O(1/n)$. For $d = 1$, bounded variation in the sense of Hardy and Krause reduces to the familiar one dimensional notion of bounded variation. As a result a log power $r > 0$ can apply to the limit as $n \to \infty$ through positive integers but not as $n = b^m$ for $m \to \infty$.

5 Empirical Investigations for $d = 2$

This section reports on an empirical search for an integrand with errors that are $\Omega(\log(n)^r/n)$ for some $r > 1$ when using a sequence of points with $D^*_n = O(\log(n)^2/n)$. We know from Sect. 3 that this is attainable for $1 < r < 3/2$ and not ruled out for $3/2 \leqslant r < 2$. We search using some knowledge of the weaknesses of the van der Corput points for the functions from Sect. 4.

The search must take place with $d \geqslant 2$ and so $d = 2$ is the natural first place to look for a problematic integrand. It is clear that the integrand should not be additive because then integrating it involves a sum of two one dimensional integration problems where we know that $r > 1$ is not needed. We look at two generalizations of the van der Corput sequence. The first is the Halton sequence for $d = 2$. In that sequence, $\boldsymbol{x}_{i1}$ is the van der Corput sequence in base 2 and $\boldsymbol{x}_{i2}$ is the van der Corput sequence in base 3. The second sequence is the Sobol' sequence in base 2. It has $t = 0$ and like the Halton points, the first component $\boldsymbol{x}_{i1}$ is the van der Corput sequence in base 2. For $n = 2^m$, the second component $\boldsymbol{x}_{i2}$ of the Sobol' sequence is a permutation of the first n elements of the van der Corput sequence.

We look first at the scaled error $n|\hat{\mu}_n - \mu|$ for $f(\boldsymbol{x}_i) = (x_{i1} - 1/2)(x_{i2} - 1/2)$. This integrand has integral zero and two dimensional Vitali variation of one. It integrates to zero over each component of $\boldsymbol{x}$ when the other component is fixed at any value. It has no nonzero ANOVA component of dimension smaller than 2. Each of the factors is a persistent fooling function for the van der Corput points.

For the Halton sequence, the scaled error equals $1/4$ when $n = 1$ and it remains below $1/4$ for $2 \leqslant n \leqslant 2^{20}$. It repeatedly approaches $1/4$ from below. Figure 2 shows the scaled errors for this product integrand for both Halton and Sobol' points for $n < 2^{19}$. Every 10'th value is shown to control the file size. For both constructions we see no evidence that the error is $\Omega(\log(n)^r/n)$ for $r > 1$. In fact, what is quite surprising there is the apparent $O(1/n)$ error rate, which is then *better* than what the van der Corput sequence attains for $f(x) = x$.

By plotting only 2^{19} points instead of all 2^{20} it becomes possible to see something of the structure within the triangular peaks for the Sobol' points. The linear scale for n (versus a logarithmic one) shows a clear repeating pattern consistent with an $O(1/n)$ error rate. Whether or not the errors are $O(1/n)$, this integrand is not a promising one to investigate further in the search for an integrand needing $r > 1$.

Another challenging integrand for van der Corput points was $\mathbf{1}\{x < 2/3\}$. For the Sobol' points we then take $f(\boldsymbol{x}) = \prod_{j=1}^{2}(\mathbf{1}\{x_j < 2/3\} - 2/3)$ for $d = 2$. Once again we have removed the additive contribution to the integrand that we know is integrated at the $\log(n)/n$ rate making it easier to discern the effect of the bivariate

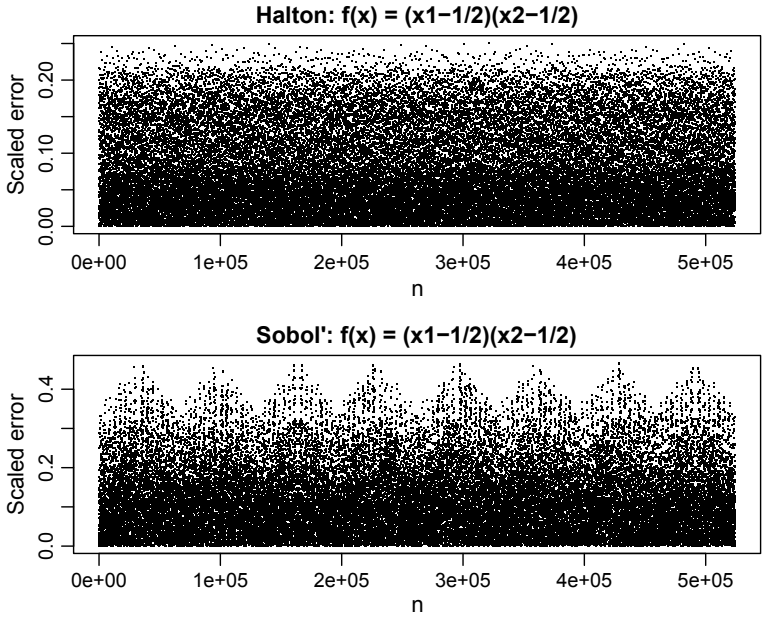


Fig. 2 The panels show scaled errors $n|\hat{\mu}_n - \mu|$ versus n for the integrand $f(\boldsymbol{x}) = (x_1 - 1/2)(x_2 - 1/2)$ on $[0, 1]^2$ and every tenth n. The top panel is for Halton points and the bottom one is for Sobol' points

component. For Halton points, we do not use $2/3$ as that has a terminating expansion in base 3 that defines the second component of the $\boldsymbol{x}_i$. We use $f(\boldsymbol{x}) = (\mathbf{1}\{x_1 < 2/3\} - 2/3)(\mathbf{1}\{x_2 < 3/5\} - 3/5)$ for Halton points because $3/5$ does not have a terminating expansion in base 3 that could make the problem artificially easy. Both of these functions have two dimensional Vitali variation equal to one, just like $\prod_{j=1}^{2}(x_j - 1/2)$. Figure 3 shows the scaled errors $n|\hat{\mu} - \mu|/\log(n)$. The logarithmic scaling in the denominator is there so that a value of $r > 1$ would give an infinite series of new records. We don't see many such new records for $n \leqslant 2^{22}$ for either of the two test cases. These integrands were designed to exploit weaknesses in the Sobol' and Halton sequences and they did not produce the appearance of errors growing faster than $\log(n)/n$. It remains possible that errors grow like $\log(n)^r/n$ for $r > 1$ but with the records being set very sparsely. The situation is quite different from Fig. 1 in the one dimensional case where we see a steady sequence of record values with a fractal pattern in the empirical errors.

There are some other integrands that could be difficult for QMC. A badly approximable irrational number θ is one where the distance between $n\theta$ and $\mathbb{Z}$ is above c/n for some $c > 0$ and all integers $n \geqslant 1$. For instance, $\theta = \sqrt{2} - 1$ is badly approximable. An integrand like $\prod_{j=1}^{d}(1_{x_j < \theta} - \theta)$ could pose a challenge to QMC methods, but some exploration with Halton and Sobol' points was inconclusive: there was no empirical indication that $r > 1$ would be required. Another potentially difficult inte-

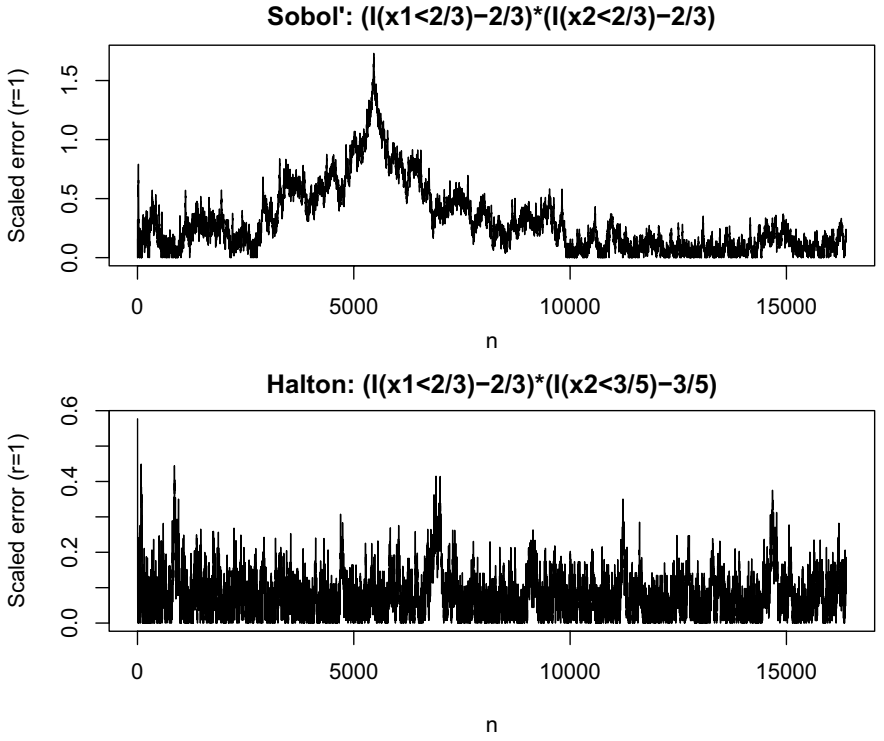


Fig. 3 The panels show scaled errors $n|\hat{\mu}_n - \mu|/\log(n)$ versus $n > 1$. The top panel has $f(\boldsymbol{x}) = (\mathbf{1}\{x_1 < 2/3\} - 2/3) \times (\mathbf{1}\{x_2 < 2/3\} - 2/3)$ for Sobol' points. The bottom panel has $f(\boldsymbol{x}) = (\mathbf{1}\{x_1 < 2/3\} - 2/3) \times (\mathbf{1}\{x_2 < 3/5\} - 3/5)$ for Halton points

grand is $\prod_{j=1}^{d}(x_j^{\theta} - 1/(1+\theta))$ for $0 < \theta < 1$. Partial derivatives of this integrand with respect to a subset of components x_j are unbounded. That would make them fail the sufficient conditions for scrambled nets to attain $O(n^{-3/2+\epsilon})$ RMSE in [16, 19, 28]. These integrands did not show a need for $r > 1$.

Next, we consider the function $f(\boldsymbol{x}) = \mathbf{1}\{x_1 + x_2 < 1\}$. This integrand has infinite Vitali variation over $[0, 1]^2$. Therefore it also has infinite Hardy-Krause variation and so the Koksma-Hlawka bound degenerates to $+\infty$. Because f is Riemann integrable we know that $\hat{\mu}_n \to \mu_n$ if $D_n^* \to 0$. Finding that $r > 1$ for this f would not provide an example of a BVHK function needing that $r > 1$. It remains interesting to see what happens because the case is not covered by QMC theory without randomization.

Figure 4 shows scaled errors for this integrand, using an exponent of $r = 1$ for $\log(n)$. Every tenth value is used to control the file size. There is a striking difference between the results for Sobol' points versus Halton points. We see a few approximate doublings of the scaled error for Sobol' points even when using $r = 1$. This does not prove that we need $r > 1$ here; perhaps we saw the last doubling or perhaps similar jumps for larger n arise at extremely sparse intervals. It does serve to raise some

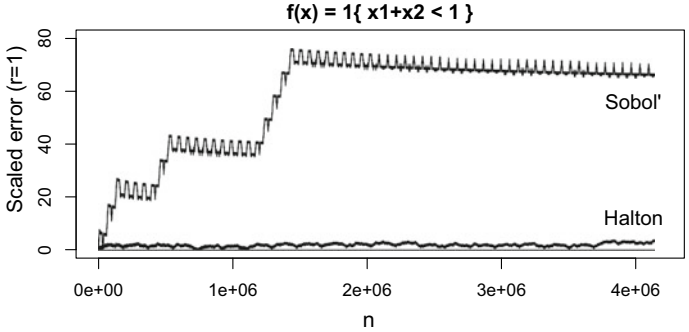


Fig. 4 This shows scaled errors $n|\hat{\mu}_n - \mu|/\log(n)$ versus $n > 1$ for every tenth n. The integrand is $\mathbf{1}\{x_1 + x_2 < 1\}$ which has infinite Vitali variation on $[0, 1]^2$. The upper curve is for Sobol' points. The lower curve is for Halton points. There is a reference line at scaled error of zero

additional questions. For instance, why are Halton points so much more effective on this integrand, and to which other integrands of unbounded variation might that apply? For smooth integrands, Sobol' points commonly perform much better when n is a power of two than otherwise. Here, those sample sizes did not bring much better outcomes.

6 Very Large *m* for Sobol' Nets

If the need for $r > 1$ is only evident for very large n then we might fail to detect it by computing $\hat{\mu}_n$. If we restrict to sample sizes $n = 2^m$ for $m \geqslant 0$ then we can use properties of the generating matrices of Sobol' sequences to compute the scaled error $n|\hat{\mu} - \mu|$ for very large n when f is the indicator function of a set $[\mathbf{0}, \boldsymbol{a})$. The first $n = 2^m$ Sobol' points form a (t, m, s)-net in base 2 for which the Koksma-Hlawka bound gives a rate of $O(n^{-1}\log(n)^{d-1})$. As a result we must look to $d = 3$ or larger for a problem that needs $r > 1$ for these values of n. We choose $\boldsymbol{a} = (2/3, 2/3, \ldots, 2/3)$ of length d as $2/3$ is difficult to approximate in base 2.

We can partition $[0, 2/3)^d$ into a countable number of elementary intervals in base 2. The number of points of the digital net $\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{b^m-1}$ that are in $E(\boldsymbol{k}, \boldsymbol{c})$ equals the number of solutions $\mathbf{i} \in \{0, 1\}^m$ to a linear equation $C\mathbf{i} = \mathbf{a} \bmod 2$ for a matrix $C \in \{0, 1\}^{|\boldsymbol{k}|\times m}$ that takes the first k_j rows from the j'th generating matrix of the digital net, for $j = 1, \ldots, d$ and some $\mathbf{a} \in \{0, 1\}^m$. See [20] for a description and a discussion of how to find the number of such points. For our Sobol' points we use the direction numbers from Joe and Kuo [11].

To make the computation finite, we replace $[0, 2/3)^d$ by $[0, a_m)^d$ where $a_m = \lfloor 2^m(2/3)\rfloor/2^m$. For $a \in (0, 1)$, let $\mu(a) = a^d$ and $\hat{\mu}(a) = (1/n)\sum_{i=0}^{n-1} 1_{\boldsymbol{x}_i \in [0,a)^d}$. Then

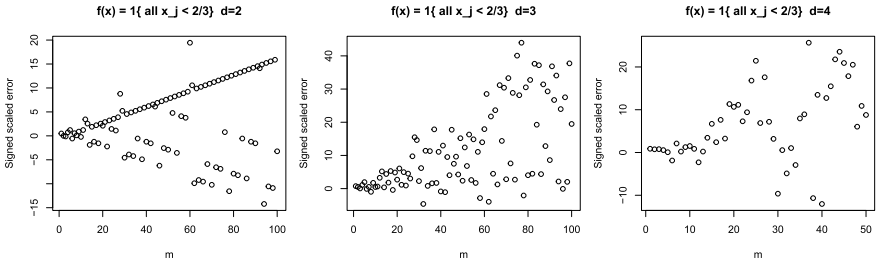


Fig. 5 The panels show the signed scaled error $n(\hat{\mu} - \mu)$ for a Sobol' sequence when $f = \prod_{j=1}^{d} 1_{x_j<2/3}$. The panels have $d = 2, 3, 4$. The sample sizes go to 2^{100} for $d = 2, 3$ and to 2^{50} for $d = 4$. The computed values differ from the true ones by at most d for all m

$$0 \leqslant \mu\Big(\frac{2}{3}\Big) - \mu(a_m) = \Big(\frac{2}{3} - a_m\Big) \sum_{j=0}^{d-1} \Big(\frac{2}{3}\Big)^j a_m^{d-j-1} \leqslant \Big(\frac{2}{3}\Big)^{d-1} \frac{d}{n}.$$

The number of the first n Sobol' points that belong to $[0, 2/3)^d \setminus [0, a_m)^d$ is at most d. Therefore

$$0 \leqslant \hat{\mu}(2/3) - \hat{\mu}(a_m) \leqslant d/n.$$

Now

$$\begin{aligned} &n(\hat{\mu}(2/3) - \mu(2/3)) - n(\hat{\mu}(a_m) - \mu(a_m)) \\ &= n(\hat{\mu}(2/3) - \hat{\mu}(a_m) - n(\mu(2/3) - \mu(a_m)) \\ &\in [-(2/3)^{d-1}d, d] \subseteq [-4/3, d] \end{aligned}$$

so the absolute error in using $n((\hat{\mu}(a_m) - \mu(a_m))$ instead of $n((\hat{\mu}(2/3) - \mu(2/3))$ is at most d.

Figure 5 shows the results for $d = 2, 3, 4$. For $d = 2$ we see strong linear trend lines in the scaled error consistent with needing $r = 1$. For $d = 3, 4$ the trend is not obviously linear but it does not have a compelling and repeating $r > 1$ pattern even at $n = 2^{50}$ for $d = 4$ or $n = 2^{100}$ for $d = 3$.

7 Discussion

We have shown that for any $\epsilon > 0$ and any $d \geqslant 1$ and any infinite sequence of points $\boldsymbol{x}_i \in [0, 1]^d$ and any $c > 0$ there exist functions in BVHK$[0, 1]^d$ where the QMC error exceeds $c(\log n)^r/n$ infinitely often when $r < (d - 1)/2$. For $d = 1$ we can easily construct such functions for $r = 1$ using results from discrepancy theory. We don't know of any specific examples with $r > 1$ and simple multidimensional

generalizations of the functions needing $r = 1$ for $d = 1$ did not show an apparent need for $r > 1$ when $d = 2$. The best candidates we saw for the Sobol' sequence are $f(\boldsymbol{x}) = \prod_{j=1}^{d} \mathbf{1}\{x_j < 2/3\}$ for $d = 3$ or 4 but we have no proof that they require $r > 1$.

One surprise is that comparing Figs. 1 and 2 we see an error for $f(\boldsymbol{x}) = (x_1 - 1/2)(x_2 - 1/2)$ that appears to be $O(1/n)$ while the one dimensional function $f(x) = x$ has error $\Omega(\log(n)/n)$ (theoretically and also empirically over small n). A second surprise is that for a two dimensional function of unbounded variation we see very different behavior for Sobol' and Halton points in Fig. 4. The error for Sobol' points appears to grow faster than $\log(n)/n$ while that for Halton points is far lower and might even grow at a different rate. Neither of these surprises contradict known theory but it is odd to see the two dimensional problem apparently easier to solve than a corresponding one dimensional one and it is also odd to see Halton points appear to be so much more robust to unbounded variation than Sobol' points.

So, where could the logs be? It is possible that they are only practically important for enormous n, or what is almost the same thing, that they are present with a very tiny implied constant. It is also possible that the commonly investigated integrands have error $O(\log(n)/n)$ even for $d \geqslant 2$.

Acknowledgements This paper is dedicated to Pierre L'Ecuyer on the occasion of his 70th birthday. Pierre has made singular contributions to Monte Carlo and quasi-Monte Carlo. His influence goes well beyond establishing theoretical results by also providing computational tools, definitive survey articles, and papers with exemplary applications, on top of service to those fields through editorial work and conference organization.

This work was supported by the U.S. NSF under grant IIS-1837931. Thanks to Fred Hickernell and Erich Novak for discussions related to this problem. We are also grateful to Traub, Wasilkowski and Woźniakowski for ensuring that Trojan's work was not lost. Finally we thank two reviewers for their helpful comments. Finally, we thank the festschrift organizers, Bruno Tuffin, Christiane Lemieux, Alex Keller and Zdravko Botev for organizing this project.

References

1. Bratley, P., Fox, B.L., Niederreiter, H.: Implementation and tests of low-discrepancy sequences. ACM Trans. Model. Comput. Simul. (TOMACS) **2**(3), 195–213 (1992)
2. Chen, W.W.L., Travaglini, G.: Some of Roth's ideas in discrepancy theory. In: Analytic Number Theory: Essays in Honour of Klaus Roth, pp. 150–163. Cambridge University Press, Cambridge, UK (2009)
3. Dick, J., Sloan, I.H., Wang, X., Woźniakowski, H.: Liberating the weights. J. Complex. **20**(5), 593–623 (2004)
4. Dick, J., Hinrichs, A., Pillichshammer, F.: Proof techniques in quasi-Monte Carlo theory. J. Complex. **31**(3), 327–371 (2015)
5. Drmota, M., Larcher, G., Pillichshammer, F.: Precise distribution properties of the van der Corput sequence and related sequences. Manuscripta Mathematica **118**(1), 11–41 (2005)
6. Faure, H.: Discrépance de suites associées à un système de numération (en dimension s). Acta Arithmetica **41**, 337–351 (1982)
7. Faure, H., Lemieux, C.: A variant of Atanassov's method for (t, s)-sequences and (t, e, s)-sequences. J. Complex. **30**(5), 620–633 (2014)

8. Halasz, G.: On Roth's method in the theory of irregularities of point distributions. Recent Progress in Analytic Number Theory, vol. 2, pp. 79–94 (1981)
9. Halton, J.H.: On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. Numerische Mathematik **2**(1), 84–90 (1960)
10. Hickernell, F.J.: Koksma-Hlawka Inequality. Wiley StatsRef: Statistics Reference Online (2014)
11. Joe, S., Kuo, F.Y.: Constructing Sobol' sequences with better two-dimensional projections. SIAM J. Sci. Comput. **30**(5), 2635–2654 (2008)
12. L'Ecuyer, P.: Randomized quasi-Monte Carlo: an introduction for practitioners. In: Owen, A.B., Glynn, P.W. (eds.) International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing, pp. 29–52. Springer (2018)
13. L'Ecuyer, P., Lemieux, C.: A survey of randomized quasi-Monte Carlo methods. In: Dror, M., L'Ecuyer, P., Szidarovszki, F. (eds.) Modeling Uncertainty: An Examination of Stochastic Theory, Methods, and Applications, pp. 419–474. Kluwer Academic Publishers (2002)
14. Niederreiter, H.: Point sets and sequences with small discrepancy. Monatshefte fur mathematik **104**(4), 273–337 (1987)
15. Niederreiter, H.: Random Number Generation and Quasi-Monte Carlo Methods. SIAM, Philadelphia, PA (1992)
16. Owen, A.B.: Scrambled net variance for integrals of smooth functions. Ann. Stat. **25**(4), 1541–1562 (1997)
17. Owen, A.B.: Scrambling Sobol' and Niederreiter-Xing points. J. Complex. **14**(4), 466–489 (1998)
18. Owen, A.B.: Multidimensional variation for quasi-Monte Carlo. In: Fan, J., Li, G. (eds.) International Conference on Statistics in Honour of Professor Kai-Tai Fang's 65th Birthday (2005)
19. Owen, A.B.: Local antithetic sampling with scrambled nets. Ann. Stat. **36**(5), 2319–2343 (2008)
20. Pan, Z., Owen, A.B.: The nonzero gain coefficients of Sobol's sequences are always powers of two. Technical Report arXiv:2106.10534, Stanford University (2021)
21. Roth, K.F.: On irregularities of distribution. Mathematica **1**(2), 73–79 (1954)
22. Schlier, Ch.: Error trends in quasi-Monte Carlo integration. Comput. Phys. Commun. **159**(2), 93–105 (2004)
23. Schürer, R., Schmid, W.C.: MinT-new features and new results. In: L'Ecuyer, P., Owen, A.B. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2008, pp. 501–512. Springer, Berlin (2009)
24. Sobol', I.M.: The distribution of points in a cube and the accurate evaluation of integrals. USSR Comput. Math. Math. Phys. **7**(4), 86–112 (1967)
25. Traub, J.F., Wasilkowski, G.W., Woźniakowski, H.: Information-Based Complexity. Academic Press, San Diego, CA (1988)
26. van der Corput, J.G.: Verteilungsfunktionen I. Nederl. Akad. Wetensch. Proc. **38**, 813–821 (1935)
27. Woźniakowski, H.: Average case complexity of multivariate integration. Bull. (New Series) Am. Math. Soc. **24**(1), 185–194 (1991)
28. Yue, R.X., Mao, S.S.: On the variance of quadrature over scrambled nets and sequences. Stat. Probab. Lett. **44**(3), 267–280 (1999)

Network Reliability, Performability Metrics, Rare Events and Standard Monte Carlo

Gerardo Rubino

Abstract In this chapter, we consider static models in network reliability, that cover a huge family of applications, going way beyond the case of networks of any kind. The analysis of these models is in general #P-complete, and Monte Carlo remains the only effective approach. We underline the interest in moving from the typical binary world where components and systems are either up or down, to a multivariate one, where the up state is decomposed into several performance levels. This is also called a performability view of the system. The chapter then proposes a different view of Monte Carlo procedures, where instead of trying to reduce the variance of the estimators, we focus on their time complexities. This view allows a first straightforward way of exploring these metrics. The chapter focuses on the resilience, which is the expected number of pairs of nodes that are connected by at least one path in the model. We discuss the ability of the mentioned approach for quickly estimating this metric, together with variations of it. We also discuss another side effect of the sampling technique proposed in the text, the possibility of easily computing the sensitivities of these metrics with respect to the individual reliabilities of the components. We show that this can be done without a significant overhead of the procedure that estimates the resilience metric alone.

Keywords Network reliability · Network performability · Rare events · Monte Carlo · Sensitivity analysis

1 Introduction

Network reliability refers to a large family of random graph-based models used in multiple areas of science and technology, for different types of analysis [2]. The reference case, and the one with which we will work in this chapter, is when we use them to analyze a communication network composed of a set $\mathcal{V}$ of nodes connected by a set $\mathcal{E}$ of edges (undirected links, no loops). The graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is called the

G. Rubino (✉)
INRIA, Campus de Beaulieu, Rennes, France
e-mail: Gerardo.Rubino@inria.fr

Z. Botev et al. (eds.), *Advances in Modeling and Simulation*,
https://doi.org/10.1007/978-3-031-10193-9_20

underlying graph in the model. We are then interested in the communications between nodes using paths. Graph $\mathcal{G}$ is assumed to be connected. With each link i we associate a given fixed probability r_i, and the interpretation is that with probability $1 - r_i$ the link is removed from the graph. More formally, with each link i we associate a Bernoulli or Binary random variable (r.v.) X_i with parameter r_i; this means that $X_i \in \{0, 1\}$ and that $r_i = \mathbb{P}(X_i = 1)$. We call r_i the *(elementary) reliability* of link i and X_i the *state* of link i. Observe that there is no explicit time variable. That is why we call these models *static* [17]. We see the system at a "typical" point in time, or at a specific instant of interest, possibly at infinity, but there is no stochastic process in the model. Denote by $m = |\mathcal{E}|$ the number of edges. Then, and always referring to the reference model, we assume that the m r.v.s $X_1, \ldots, X_m$ are independent.

With this setting, we build a random graph G with values in the set of partial subgraphs of $\mathcal{G}$ (that is, G has the same nodes as $\mathcal{G}$ and a subset of $\mathcal{G}$'s edges). Observe that there are 2^m points in G's space of values. Informally, in a realization of G edge $i \in \mathcal{E}$ exists with probability r_i. More formally, we define the distribution of G by the following expression: for any subset E of $\mathcal{E}$,

$$\mathbb{P}\big(G = (\mathcal{V}, E)\big) = \Pi_{i \in E} r_i \Pi_{j \notin E} (1 - r_j).$$

Using this random object, we can define many *connectivity*-based metrics quantifying different properties of G related to its capability to transport something (information, some fluid, …) from some nodes to other nodes in the model. By far, the most used one is the so-called *source-to-terminal reliability* defined as follows: two different nodes are fixed, say s and t, and the metric is $R_{s,t} = \mathbb{P}(s \text{ and } t \text{ are connected in } G)$, that is, the probability that s and t belong to the same connected component of G. This is also called the *2-terminal* reliability, or simply the *s-t*-reliability. Another widely used metric is the *all-terminal* reliability, R_{all}, defined by $R_{\text{all}} = \mathbb{P}(\text{all nodes are connected in } G)$, that is, the probability that G is connected (that it has a single connected component). Formally, we usually code the fact that the selected property holds by 1 and that it doesn't hold by 0, and we call *system's state* that random number.

There are many variations around this melody. Instead of looking at the connections between two nodes s and t or between all pairs of nodes, we can define a K-terminal reliability R_K concerning a given subset K of nodes; the corresponding definition is

$$R_K = \mathbb{P}(\text{all nodes in } K \text{ belong to the same connected component of } G).$$

The connectivity criteria can be more complicated. For instance, we can define the *distance-based 2-terminal reliability with parameter d*, $R_{d;s,t}$, where $d \geq 1$ is an integer, by

$$R_{d;s,t} = \mathbb{P}(s \text{ and } t \text{ are connected in } G \text{ by at least a path having length } \leq d),$$

and similarly for R_{all} or R_K [6]. A different extension consists of associating a non-negative integer k_s with node s for all $s \in \mathcal{V}$ and defining a new metric as the probability that there are $\min\{k_s, k_t\}$ elementary edge-disjoint paths between nodes s and t, for all s and t [12]. All these metrics can also be defined in a similar model where the nodes and not the edges, have random states, or both nodes and edges are random, on directed graphs, instead of undirected ones, etc. Recently, some works appeared which remove the independence assumption concerning the state variables [3, 21].

The evaluation of all these reliability metrics is, in general, a computational hard problem [22]. Formally, it belongs to the $\#P$–complete class, a family of NP–hard problems not known to be in NP. A #P-complete problem is equivalent to counting the number of solutions to an NP-complete one. For this reason, $\#P$–complete problems are, at least, as hard as NP–complete ones. Even in very restricted classes of graphs, the computation of the before mentioned metrics remains in this complexity class. For instance, it is shown in [16] that the computation of the source-to-terminal reliability is in the $\#P$–complete class even when the graph is planar (in fact, s, t-planar) and has vertex degrees at most equal to three. From a practical point of view, this means that a graph with, say, more than one hundred elements (nodes, lines) can not be exactly evaluated (except for special cases). An efficient alternative is then to use a Monte Carlo method, the topic of this chapter. The standard version consists of generating an N-sample of G, say $G_1, \ldots, G_N$, and to estimate the reliability metric, say $R_{s,t}$, using the standard estimator

$$\widehat{R}_{s,t} = \frac{1}{N} \sum_{n=1}^{N} 1\big(s \text{ and } t \text{ connected in } G_n\big), \tag{1}$$

where $1(P)$ is equal to 1 if the predicate P is true, 0 if it is false. In many cases, this approach can handle medium and large size models, but, of course, it has its own problem, namely the fact that its efficiency is sensitive to the numerical values of data (the r_is). In particular, this standard procedure (sampling N times graph G and checking the property used to define the desired metric on each realization) becomes of no use in the *rare event* case, that is, when the system reliability metric is (very) close to one. Unfortunately, this is often the interesting case.

For the reader unfamiliar with the area, let us finally comment that the specific source-to-terminal model has a huge application area in the dependability analysis of many kinds of systems. The general literature is rich in examples. Let us just mention [20], where bounds of the metric $R_{s,t}$ are proposed, applied to the analysis of critical control subsystems of a modern aircraft.

There is a very large literature in the network reliability area. A good review of the main concepts and results is [2], and an older and classical text, [14]. Another already mentioned survey is ours [17]. Specifically in the Monte Carlo side, some papers by Pierre L'Ecuyer and co-authors are [4, 5, 13]. See also [19] on the rare event side, in particular Chap. 7, or [8] for many technical aspects.

In the sequel of this chapter, we explore in Sect. 2 the interest of using *performability* metrics to quantify the capacity of a network to provide service in spite of failures (or equivalent events) of its components, and we focus on one of them, called *resilience*. Then, in Sect. 3, we discuss the estimation of the resilience using the naive Monte Carlo approach. After commenting the rare event problem, we show that it is possible to be efficient in the use of the naive, or standard, or crude estimator, by means of a smarter implementation (this extends old work presented in [11]). In Sect. 4, we illustrate the use of the previously discussed implementation, and we show that it allows exploring other types of metrics, written as variations on the resilience idea. We also show that the same approach makes straightforward to evaluate also the gradient of the resilience with respect to the elementary reliabilities, with almost no overhead (extension to ideas presented in [18]). Section 5 concludes the chapter.

2 Performability Metrics and Resilience

Quantitative analysis of systems is mainly done following two different viewpoints. Either we assume the system perfect (that is, we ignore its possible failures and repairs) and we focus on measures related to the work it does, on the properties of the service it provides, or we ignore the latter and we focus on those possible failures and repairs, and try to analyze different related properties of the system. Around the 80s appeared another idea, in general a much more complex one, where we look simultaneously, that is, in the same model, at both sides of systems' behavior. Following the pioneering work done by Meyer [15], we call it *performability*. We will consider this view here, based on the reference model presented before.

So, we have an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ assumed to be connected and with no loops, with n nodes and m edges (the underlying graph), and our standard probabilistic setting parameterized by the elementary reliabilities $r_1, \ldots, r_m$. Consider now the R_{all} metric, widely used to quantify the ability of the network to support communications between all pairs of nodes. It measures a binary property, an all-nothing one, the probability that all nodes can communicate. But for the system manager, or for the users, the fact that, for instance, a single node is isolated, making that among the total number $n(n-1)/2$ of pairs of nodes, only $n-1$ of them can't communicate, is very different than the situation where many links are failed making that just a few pairs of nodes can talk to each other (possible none of them if all links are failed). A way of measuring this is by directly looking at the number of pairs that can communicate.

We thus define the central r.v. of this chapter,

$$NCP = \text{number of pairs of nodes that can communicate in } G.$$

The analysis of this r.v. provides a much richer information about the capacity of the network to provide communications between its nodes. We will focus on its expectation in the sequel.

2.1 The Resilience Metric

The expectation of *NCP* has been called *resilience* in some works [23], in the same context as in this chapter. It has been explored in a few other works [1, 9], always in the setting discussed here. So, formally, the resilience *Res* of our model is

$$Res = \mathbb{E}(NCP). \tag{2}$$

Observe first that we have the following immediate properties:

- $0 \le NCP \le \binom{n}{2} = \dfrac{n(n-1)}{2}$;
- $\mathbb{P}(NCP = 0) = \Pi_{i \in \mathcal{E}}(1 - r_i)$;
- $\mathbb{P}\left(NCP = \binom{n}{2}\right) = R_{\text{all}}$;
- $Res|_{\forall i \text{ in } \mathcal{E},\, r_i = 0} = 0, \qquad Res|_{\forall i \text{ in } \mathcal{E},\, r_i = 1} = \binom{n}{2}$.

We can normalize the metric, dividing $\mathbb{E}\big(NCP\big)$ by $\binom{n}{2}$, thus leading to an index in [0, 1]. The *scaled resilience* of the network, *ResScaled*, is then $ResScaled = 2Res/\big(n(n-1)\big)$.

Let us provide first a few simple examples, assuming, to simplify the presentation, that we are in the i.i.c. case, meaning independent and identical components (here, we refer to the links) with common elementary reliability r.

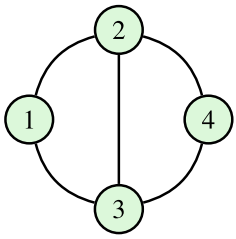
Bridge. Take first a bridge, as depicted in Fig. 1. Using brute force (listing all 2^5 possible links' states), we get $Res = r\big(5 + 8r - 14r^3 + 7r^4\big)$.

Paths. Consider now a path with n nodes (thus, with $n - 1$ edges), $n \ge 2$. Denote by Res_n its resilience. Conditioning on the state of any of the 2 extreme edges, we can verify, after some algebra, that

$$Res_n = \frac{r\big[n(1-r) - (1-r^n)\big]}{(1-r)^2} = \frac{r\big[n - 1 - r(n - r^{n-1})\big]}{(1-r)^2}.$$

A few first values: ($Res_1 = 0$,) $Res_2 = r$, $Res_3 = 2r + r^2$, $Res_4 = 3r + 2r^2 + r^3$.

Fig. 1 The bridge topology



Rings. Last, consider a ring with $n \geq 3$ nodes (or $n \geq 2$ if we accept the multigraph corresponding to the $n = 2$ case). Denote again Res_n its resilience. Conditioning on the state of any of its links, we obtain

$$Res_n = nr\left(\frac{1-r^{n-1}}{1-r} - \frac{n-1}{2}r^{n-1}\right).$$

A few first values of Res_n: ($Res_1 = 0$,) $Res_2 = 2r - r^2$, $Res_3 = 3r + 3r^2 - 3r^3$, $Res_4 = 4r + 4r^2 + 4r^3 - 6r^4$.

2.2 Some Properties of Resilience

The "first moment principle" says that if $X \geq 0$ is an integer r.v., then it holds that $\mathbb{P}(X > 0) \leq \mathbb{E}(X)$. Applied to $\mathbb{E}\big(NCP\big)$, this means that $1 - \Pi_{i\in\mathcal{E}}(1-r_i) \leq Res$. However, this is not very good, because Res is, in general, > 1, and $1 - \Pi_{i\in\mathcal{E}}(1-r_i)$ is ≤ 1.

Now, simply by definition, $\binom{n}{2}R_{\text{all}} \leq Res \leq \binom{n}{2}$, which are better bounds, especially in the rare event case. For the scaled resilience, we have $R_{\text{all}} \leq ResScaled \leq 1$.

Now, define the r.v.

$$Y_{s,t} = 1\big(\text{there is a path connecting nodes } s \text{ and } t\big).$$

We have $R_{s,t} = \mathbb{P}(Y_{s,t} = 1) = \mathbb{E}(Y_{s,t})$ and also

$$NCP = \sum_{\text{all nodes } s,t,s<t} Y_{s,t},$$

from which, taking expectations, we have the following important relation, making the connection between this performability approach and the classical binary metrics:

$$Res = \sum_{\text{all nodes } s,t,s<t} R_{s,t}. \tag{3}$$

Conditioning with respect to the state X_i of any edge i, we can write

$$Res(\mathcal{G}) = \mathbb{E}(NCP(\mathcal{G}) \mid X_i = 1)r_i + Res(\mathcal{G}_i^d)(1-r_i), \tag{4}$$

where we use the notation $Res(H)$ to make explicit the fact that we define the metric on graph H, and where $\mathcal{G}_i^d$ is the graph obtained by removing edge i from $\mathcal{G}$. If $\mathcal{G}_i^c$ denotes the graph obtained by contracting edge i in $\mathcal{G}$, here we don't have the usual fact that $Res(\mathcal{G}_i^c)$ coincides with $\mathbb{E}(NCP(\mathcal{G}) \mid X_i = 1)$, a relation that holds for classical metrics such as $R_{s,t}$. This makes that Relation (4) is less useful than for

standard metrics, where it is the basis of the most powerful approach for their exact numerical analysis. This is because of the more complex relation between $Res(\mathcal{G}_i^c)$ and $Res(\mathcal{G})$. This issue is out of the scope of this paper, so, we will not pursue commenting it here.

In the rest of this paper, we will look at the evaluation of this expectation using Monte Carlo.

3 Using Standard Monte Carlo for Resilience-Based Analysis

First of all, consider reliability metrics such as $R_{s,t}$ or R_{all} (or any of their many variations). Since these metrics are typically close to 1, it is better to work with the complementary events and target the corresponding unreliability values, which we will denote here by γ (for instance, $1 - R_{s,t}$, or $1 - R_{\text{all}}$). The fact that now the target γ is close to 0 makes clear that the correct way to look at the quality of these estimations is through the analysis of relative errors and not absolute ones.

3.1 The Standard Estimator

Estimating an unreliability metric such as $1 - R_{s,t}$ or $1 - R_{\text{all}}$, say generically $\gamma = 1 - R$, and using the standard or crude estimator, means sampling the following r.v.:

$$\widehat{\gamma} = \frac{1}{N} \sum_{n=1}^{N} Y^{(n)},$$

where $Y^{(1)}, \ldots, Y^{(N)}$ are N independent copies of the Binary r.v. Y equal to 1 if the considered connectivity property doesn't hold, and to 0 otherwise (Y is the complementary of what we called the system's state). We have $\mathbb{E}(\widehat{\gamma}) = \mathbb{E}(Y) = \gamma$ (that is, $\widehat{\gamma}$ is unbiased). A pseudo-code corresponding to this estimator is as follows:

```
Algorithm A // naive implementation of standard Monte Carlo
            // goal: network unreliability
// the connectivity property used is denoted here CP
set counter to 0;
execute N times:
   sample G;
   if the CP doesn't hold: counter++
return counter/N
```

Using the states of the links $X_1, \ldots, X_m$, this means that we repeat N times the following: for each link $i \in \mathcal{E}$, we sample a Binary r.v. having parameter r_i, and we

put the result in x_i; then, we remove from the underlying graph $\mathcal{G}$ those links j such that $x_j = 0$; next step is to execute a procedure to check the connectivity property used, for instance, to check if the obtained partial graph of $\mathcal{G}$ is connected or not; at the end, we return the number of times the graph wasn't connected divided by N, which is an estimator of $\gamma = 1 - R_{\text{all}}$.

Observing first that we have $\mathbb{V}(\widehat{\gamma}) = \gamma(1-\gamma)/N$, the corresponding standard confidence interval is $\left(\widehat{\gamma} \mp z(\varepsilon)\sqrt{\widehat{\gamma}(1-\widehat{\gamma})/(N-1)}\right)$, where $1-\varepsilon$ is the confidence level, or ideal coverage probability, with

$$z(\varepsilon) = \Phi^{-1}\left(1 - \frac{\varepsilon}{2}\right), \qquad \Phi(x) = \frac{1}{\sqrt{2\pi}} \int_0^x e^{-t^2/2} \mathrm{d}t$$

(e.g., $z(0.05) \approx 1.96$ for a confidence level of 95%). The quantity $\widehat{\gamma}(1-\widehat{\gamma})/(N-1)$ is proportional to half the confidence interval length (the proportionality constant being $z(\varepsilon)$), and can thus be interpreted as an absolute error. The ratio

$$\frac{\sqrt{\dfrac{\widehat{\gamma}(1-\widehat{\gamma})}{N-1}}}{\widehat{\gamma}} = \sqrt{\frac{1-\widehat{\gamma}}{(N-1)\widehat{\gamma}}} \approx \frac{1}{\sqrt{N\widehat{\gamma}}}$$

can then be seen as an estimation of the (statistical) relative error we have, and we conclude by saying that it goes to ∞ as the rarity of the event of interest increases, that is, as $\gamma \to 0$. This formally show that as the system's failure becomes rarer, the estimation procedure becomes less accurate, and this happens without any bound.

The point to be made next is that the estimator $\widehat{\gamma}$ can be computed in a more efficient way, especially in the frequent case where the r_is are all close to 1, and this leads to efficient estimations, because what matters is to have a small product "mean running time $\times$ variance of the estimator" [7]. This is the topic developed in next subsection.

3.2 *The Standard Estimator Efficiently Implemented in the Rare Event Case*

Imagine we implement the standard estimator of γ as follows. Instead of sampling G a huge number N of times and checking the connectivity property on each sample, we sample N times each X_i and put the N results in column i of a big table. The table has $m+1$ columns, one for each link plus a last one for the complement of the system's or network's state, and N rows. Then, we read the table by rows and we fill the last $(m+1)$th column checking the system's state. Since, of course, we will not implement such a table, we will call it the *virtual* table from now on.

Now, assume for a moment that $N = \infty$, and consider column i. Since r_i is usually close to 1, the column has a majority of 1s. The position of its first 0 is a

geometric r.v. F_i with values in the strictly positive integers and distribution given by $\mathbb{P}(F_i = h) = r_i^{h-1}(1 - r_h), h \geq 1$. The discrete point process defined by the positions of the different 0s in that column is a Bernoulli process, where the distance between consecutive 0s (and from the first 0 and an auxiliary initial and empty row 0 in the table) are independent copies of F_i. Now, let $F = \min\{F_1, \ldots, F_m\}$ be the first row where there is at least one 0 in the first m columns. We have that the r.v. F is also geometric, with values on the integers ≥ 1, and law given by $\mathbb{P}(F = \ell) = q^{\ell-1}(1 - q), \ell \geq 1$, where $q = r_1 r_2 \cdots r_m$. And in the same way as for column i, if we look at those rows in the infinite virtual table where at least one link is down, the distance between consecutive rows with that property are the points of a discrete Bernoulli process with parameter q (that is, if U_k is the kth such row, then the sequence $(U_1, U_2 - U_1, U_3 - U_2, \ldots)$ is i.i.d. with the law of F. The interest of this type of description of the estimation procedure is that the statistical properties of the estimator used are those of the standard one, in particular, the variance.

This suggests a different way of implementing the standard estimator of γ. Sample F and add the obtained value to a first register *rows*. In the first $F - 1$ rows of the table, we have only 1s in the first m columns, so, the r.v.s $Y_1, \ldots, Y_{F-1}$ are 0 (technically, we assume *coherent* systems [17], so, they are always up if all the components are up). It remains to see what happens with the Fth row. For that purpose, we must sample the X_is, but now knowing that there is at least one zero in the vector of links' states. Consider the natural ordering $1, 2, \ldots, m$ of the network's links, denote by Z the r.v. "number of zeros in the vector of links' states", and by J the index of the first zero in that vector. We have then the following immediate result: for $j = 1, 2, \ldots, m$,

$$\mathbb{P}(J = j \mid Z \geq 1) = \frac{r_1 r_2 \cdots r_{j-1}(1 - r_j)}{1 - r}. \tag{5}$$

Then, we sample from this distribution, obtaining some $j \in \mathcal{E}$, we set the states of links $1, 2, \ldots, j - 1$ to 1 and that of link j to 0, and we independently sample the states of links $j + 1, \ldots, m$ using their original Binary distributions with parameters $r_{j+1}, \ldots, r_m$ respectively. Once known the states of all the links, we check if the connectivity property used holds or not. In the latter case, we increase a second register *counter*. We repeat the procedure, sampling a new value from F's law, obtaining some integer $f \geq 1$, adding it to *rows*, and repeating the conditional sampling of the links knowing that there is at least a 0 in the vector of links' states, until updating *counter*. We stop the process when we reach the condition *rows* $\geq N$, and we return the estimation *counter*$/N$. In pseudo-code, this gives Algorithm B.

```
Algorithm B // 2nd implementation of standard Monte Carlo
            // goal: network unreliability
// the connectivity property used is denoted here CP
rows = 0; counter = 0;
while rows < N: // main loop
   f = sampleFrom(F);
   rows += f;
```

```
    if rows ≤ N:
       conditionally sample vector of links' states;
       if CP doesn't hold: counter++
    // end of main loop
return counter/N
```

Recall that the instruction "`sample vector of links' states`" means sampling first J using (5), obtaining some value j, then setting $X_1 = X_2 = \cdots = X_{j-1} = 1$, $X_j = 0$, and sampling each remaining state variable X_k, for $k = j+1, \ldots, m$, according to the Binary law with parameter r_k, and independently from each other.

Complexity analysis. Consider first Algorithm A. Sampling G needs $\Theta(m)$ in computation time. Testing the connectivity property used has a cost $O(n+m)$ in most of the cases mentioned in the chapter, in particular, for the resilience (see below) and since $m \geq n-1$ because the underlying graph is connected, we can also write it $O(m)$. Since we iterate N times, the total mean computational cost is $O(Nm)$.

In Algorithm B, we will sample from F, for large N (almost always the case), on average, $N/\mathbb{E}(F)$ times. We have $\mathbb{E}(F) = (1-q)^{-1}$. The conditional sampling procedure of (5) plus the test of the selected connectivity property costs $O(m)$, so, the total average computational cost is $O(N(1-q)m)$. The time reduction in B with respect to the naive implementation of A is, then,

$$\frac{Nm}{N(1-q)m} = \frac{1}{1-q}.$$

This approach alone is modest in time reduction, but it can be improved (see below, in Sect. 3.4).

Remark 1 Observe that what we are doing is somehow inspired by a conditional Monte Carlo technique. The interest of the type of implementation of the estimation procedure followed here is that we can see it as a different algorithm for the evaluation of the standard estimator, and this can be useful, as shown below. It will be exploited for quickly exploring other possible metrics of the resilience type, and also to estimate sensitivities (gradients). The central interest in the conditioning we used is that its effect can be seen as a significant reduction in the sampling process of the standard estimator, but not on the estimator's variance, in the execution time instead.

Let us come back first to the estimation of the resilience.

3.3 Estimating the Resilience

Recall that the resilience is defined as the expected number of pairs of nodes that can communicate, in the random graph G, $\mathbb{E}(NCP)$ (Relation (2)). Observe that if the graph is connected, we have $NCP = n(n-1)/2$. Suppose now that it is not, and

let NCC be the number of connected components of G. Denote the associated node sets as $\mathcal{S}_1, \ldots, \mathcal{S}_{NCC}$ and their cardinalities by $C_1, \ldots, C_{NCC}$. We then obviously have

$$NCP = \sum_{h=1}^{NCC} \binom{C_h}{2} \tag{6}$$

where $\binom{h}{k} = 0$ when $h < k$. Since computing NCC and the cardinalities $C_1, \ldots, C_{NCC}$ needs only a DFS (Depth First Search) or a BFS (Birth First Search) [10] on G, the cost of computing NCP is $O(m)$. So, the procedure for estimating Res is basically the same in both Algorithms A and B: the only change to do is that instead of checking one of the considered connectivity properties, we have just to decompose G into its connected components and use (6). For instance, the resilience version of Algorithm B is as follows.

```
Algorithm Bbis // 2nd implementation of standard Monte Carlo
               // goal: network resilience
rows = 0;  sum = 0;
while rows < N: // main loop
   f = sampleFrom(F);
   if rows + f ≤ N:
      sum += (f − 1)n(n − 1)/2;
      conditionally sample vector of links' states with (5);
      find the connected components of G;
      sum += NCP of G using (6)
   else
      sum += (N − f)n(n − 1)/2
   rows += f
   // end of main loop
return /N
```

Formally, we are using the standard estimator

$$\widehat{Res} = \frac{1}{N} \sum_{n=1}^{N} Res^{(n)},$$

where $Res^{(n)}$ is $Res(G_n)$. A confidence interval for Res with confidence level of, say, 95%, is then given by $(\widehat{Res} \mp 1.96\, S)$, where S^2 is the standard estimator of the variance of $\widehat{Res}$. We have

$$S^2 = \frac{1}{N(N-1)} \sum_{n=1}^{N} NCP^{(n)2} - \frac{1}{N-1} \widehat{Res}^2.$$

3.4 Improving Algorithm B

Assume you know, for instance, the *breadth* b of the underlying graph, that is, the size of a mincut of minimal size (a set of edges whose removal disconnects the graph [10]). This number can be computed using a max flow algorithm such as Edmonds-Karp's, and its cost is polynomial in (n, m). For instance, suppose that $b = 2$. This means that to transform $\mathcal{G}$ into an unconnected graph by removing edges, you must remove at least 2 edges. This means that instead of checking the connectivity property used when there is at least one zero in the row of the virtual table, or instead of decomposing G into its connected components in the resilience case, we can first just count how many zeros we have, and if the number is less than b, we know that the current realization of G is connected, so, *NCP* is equal to $n(n-1)/2$. This will make a supplementary gain in performance, needing only to compute the graph breadth, which is a one shot polynomial task, executed only at the beginning. The corresponding algorithm is presented again in pseudo-code, this time for estimating the resilience.

```
Algorithm C // 3rd implementation of standard Monte Carlo
            // goal: network resilience
b = breadthOf(G);
rows = 0; sum = 0;
while rows < N: // main loop
   f = sampleFrom(F);
   if rows + f ≤ N then
      sum += (f − 1)n(n − 1)/2;
      conditionally sample vector of links' states with (5);
      if its # of zeros is ≤ b then
         sum += n(n − 1)/2
      else
         find the connected components of G;
         sum += NCP of G computed using (6)
   else
      sum += (N − f)n(n − 1)/2
   // end of main loop
return sum/N
```

For the improvement in reducing the computing time of the estimation process, let us wait until next section, where some numerical illustrations are given.

Comment. The way we presented Algorithm C is to underline that it improves version B (or Bbis). But it can be simplified, by redefining the r.v. F. Instead of being the first row of the virtual table with at least 1 zero, we can define it as the first row of the virtual table with at least b zeros. This means then to use again Algorithm B but with this new conditioning. There are other possibilities, using a covering tree of $\mathcal{G}$, or several ones, etc. To discuss the idea here we will just keep the very simple idea of counting the 0s in the vector of links' states, that is enough to describe the methodology. More on this in Sect. 4, but let us look now at sensitivities.

3.5 Sensitivity Analysis

The last element of the discussion given in this chapter concerns the sensitivity analysis of resilience. This means computing the gradient of *Res* seen as a function of the N variables $r_1, \ldots, r_m$, that is, the partial derivatives $\partial Res/\partial r_i$ for all $i \in \mathcal{E}$. This subsections builds on the results presented in [18], extended to handle the resilience concept.

Define

$$\sigma_i = \frac{\partial Res}{\partial r_i} \quad \text{and} \quad \sigma_{s,t;i} = \frac{\partial R_{s,t}}{\partial r_i}.$$

Denote, similarly as for the reliability metrics,

$$Res_i^c = Res(\mathcal{G} \mid X_i = 1) \quad (\neq Res(\mathcal{G}_i^c))$$

and

$$Res_i^d = Res(\mathcal{G} \mid X_i = 0) = Res(\mathcal{G}_i^d).$$

Then, the next result is the first step toward the estimation of the gradient of the resilience.

Proposition 1 *We have*

$$\sigma_i = \frac{Res_i^c - Res}{1 - r_i} = \frac{Res - Res_i^d}{r_i}.$$

Proof. Let us show the steps to prove the first equality. The second one is similar. In (3) we stated the basic identity

$$Res = \sum_{\text{all nodes } s,t, s<t} R_{s,t}.$$

Taking partial derivatives with respect to r_i, we get

$$\sigma_i = \sum_{\text{all nodes } s,t, s<t} \sigma_{s,t;i}.$$

In [18], we proved that

$$\sigma_{s,t;i} = \frac{R_{s,t;i}^c - R_{s,t}}{1 - r_i} = \frac{R_{s,t} - R_{s,t;i}^d}{r_i}. \tag{7}$$

Using the first equality, we have

$$\begin{aligned}\sigma_i &= \sum_{\text{all nodes } s,t,s<t} \sigma_{s,t;i} \\ &= \sum_{\text{all nodes } s,t,s<t} \frac{R^c_{s,t;i} - R_{s,t}}{1-r_i} \\ &= \frac{1}{1-r_i}\left(\sum_{\text{all nodes } s,t,s<t} R^c_{s,t;i} \;-\; \sum_{\text{all nodes } s,t,s<t} R_{s,t} \right) \\ &= \frac{Res^c_i - Res}{1-r_i}.\end{aligned}$$

The second equality follows when using the second one in (7). □

Now, in the same paper [18], it was shown that

$$\widehat{\sigma}_{s,t;i} = \frac{X_i - r_i}{r_i(1-r_i)} Y_{s,t}$$

is an unbiased estimator of $\sigma_{s,t;i}$.

Remark 2 Observe that these estimators fit precisely the idea of having at our disposal the virtual table filled with the values of the states of all links and the corresponding value of *NCP*, at each iteration of the standard method. Actually, the idea of seeing a conditional Monte Carlo approach as described here appeared at the same time that we developed the estimators of sensitivities of classical network reliability metrics of the type presented in the chapter.

Observe that evaluating $\widehat{\sigma}_{s,t;i}$ adds almost no overhead to the estimation of the reliability metrics considered (classical or of the performability type).

Based on previous result, we have the following one.

Theorem 1 *The expression*

$$\widehat{\sigma}_i = \frac{X_i - r_i}{r_i(1-r_i)} NCP$$

defines an unbiased estimator of σ_i.

Proof. In a straightforward manner,

$$\begin{aligned}\widehat{\sigma}_i &= \frac{X_i - r_i}{r_i(1-r_i)} \sum_{\text{all nodes } s,t,s<t} Y_{s,t} \\ &= \frac{1}{r_i(1-r_i)} \left(\sum_{\text{all nodes } s,t,s<t} X_i Y_{s,t} \;-\; r_i \sum_{\text{all nodes } s,t,s<t} Y_{s,t} \right).\end{aligned}$$

Taking expectations, and using the fact that $\mathbb{E}(X_i Y_{s,t}) = r_i R^c_{s,t;i}$, we get

$$\mathbb{E}(\widehat{\sigma_i}) = \frac{1}{r_i(1-r_i)} \left(\sum_{\text{all nodes } s,t,s<t} r_i R^c_{s,t;i} \quad - \quad r_i Res \right) = \frac{Res^c_i - Res}{1-r_i} = \sigma_i,$$

by means of the result of Proposition 1. □

As we can see, the estimator $\widehat{\sigma_i}$ consists basically of sampling simultaneously the states of the links and the *NCP* variable, something explicit in the virtual table. So, estimating the sensitivities of the resilience with respect to the elementary reliabilities in the system is straightforward following the same approach as for the resilience itself. Next section gives a few examples and discusses other possible resilience-like metrics easily analyzed using the procedure we have described, including the sensitivities.

4 Examples and Discussions

In this section, to simplify the presentation we will only consider the i.i.c. case (independent and identical components) in the models, that is, the case of homogeneous links, all sharing the same elementary reliability, denoted here by r.

To provide a few illustrations of previous results, let us consider the model depicted by its underlying graph in Fig. 2.

This network has $n = 21$ nodes and $m = 26$ links. Its breadth is $b = 2$. The expected reduction in the execution time, using the basic Algorithm B and written as a factor, is $1 - q$, $q = r^m = r^{26}$. Taking $r = 0.9999$ we obtain a time reduction denominator of ≈ 385. Using the breadth (Algorithm C), we have $b = 2$. The supplementary factor to add is $\mathbb{P}(Z \geq 2 \mid Z \geq 1)$, where Z denotes the number of zeros in a row, a Binomial r.v. with $\mathbb{P}(Z = z) = \binom{m}{z}(1-r)^z r^{m-z}$. The supplementary factor is then $\mathbb{P}(Z \geq 2 \mid Z \geq 1) = \mathbb{P}(Z \geq 2)/\mathbb{P}(Z \geq 1)$, and $\mathbb{P}(Z \geq 1) = 1 - \mathbb{P}(Z = 0) = 1 - q$, so, the new global average time reduction factor becomes $\mathbb{P}(Z \geq 2) = 1 - r^m - mr^{m-1}(1-r) \approx 3.2448 \cdot 10^{-6}$, which corresponds to a denominator of about 300, 000. This shows that the time reduction can be considerable even when using such a simple conditioning. The accelerations we observed in practice have

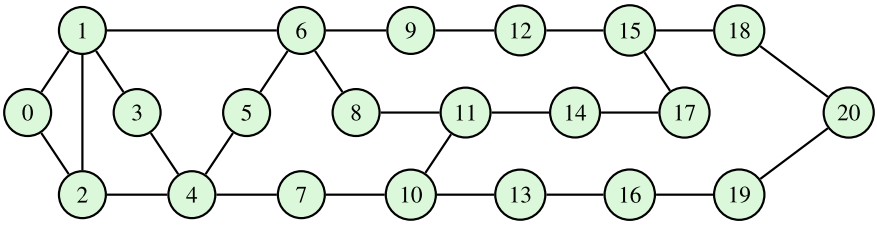


Fig. 2 A widely used Arpanet topology in network reliability, from the history of this famous communication network

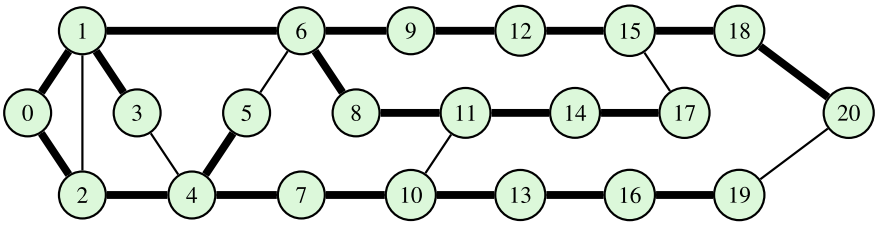


Fig. 3 A covering tree on the Arpanet model of Fig. 2

been lower than the theoretical factors, probably due to the implementations of the algorithms we had, but in all cases they were important. Using only Algorithm Bbis, in the case of $r = 0.999$, we got accelerations of several hundreds, depending on the tools used (`C` code, `python` alone, `python` with `numpy`). The accelerations were higher using the breadth, but there are other possibilities in the same spirit and still more powerful. Let us give another example here.

Assume we computed a covering tree of the underlying graph, that is, a tree sharing with the underlying graph all its nodes. If we denote the tree as $\mathcal{T} = (\mathcal{V}, \mathcal{E}')$ the probability that all its links are up is $\prod_{i \in \mathcal{E}'} r_i$. In case of that event, we know that $NCP = n(n-1)/2$, so, we can use the event to get a better reduction factor than $1 - q = 1 - r^m$. The sampling of the links' states conditioned to the fact that all links in the covering tree are working is straightforward, only the links out of the tree must be sampled. In the Arpanet model given in Fig. 2, the gain is modest. See first a covering tree of this graph in Fig. 3.

In the case of Algorithm B with $r = 0.9999$, we had a time reduction factor of about 385. Using a covering tree, always in the i.i.c. case, the factor changes to about $1 - r^{20} \approx 500$. If the graph is more dense, we can have several edge-disjoint covering trees and considering their union we can reach much higher reductions in time. We can also use mincuts with the same purpose. This will not be explored further here, we intend to show only the general principles.

In the general heterogeneous case, when the elementary reliabilities are not necessarily all equal, we can look for covering trees with minimal total weight, where the weight of the tree is *the product* of the reliabilities of its links. For that purpose, we can use standard graph algorithms such as Prim's or Kruskal's (see any good reference in the area, such as [10]). Both have complexity $O(m \ln m)$ where m is the number of edges in the graph.

Concerning sensitivities, we only illustrate the main interest in their evaluation, a point underlined already in [18]. Consider the example in Fig. 4.

We see that, from the connectivity point of view and when we have an i.i.c. model (homogenous links), the link $\{4, 5\}$ is the most important one, the first one to improve if possible. For example, in the case of $r = 0.999$, we have the estimations given in Table 1.

This illustrates well the fact that the order induced on the links by the gradient of the resilience is a much more robust result than the metric itself. Of course, the

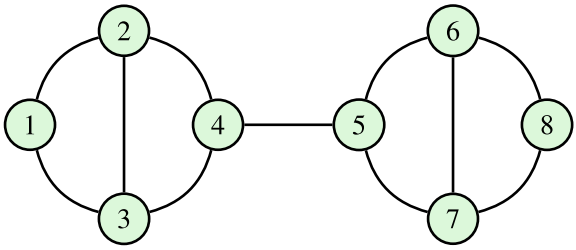


Fig. 4 Two bridges connected by a "bridge" link

Table 1 A few sensitivities of the resilience, on the model depicted in Fig. 4. In the estimations, the absolute error is less than half the last significant digit (confidence level: 95%)

edge	{1, 2}	{2, 3}	{4, 5}
sensitivity	$7.01\ 10^{-3}$	$3.79\ 10^{-5}$	16.0

Table 2 Some values of resilience as r increases. Same accuracy as in Table 1

r	0.91	0.95	0.99	0.995	0.999	0.9999
Res	199.8	207.2	209.9	209.977	209.9991	209.999989

example has been chosen to amplify the effect of having links playing a more critical role than the others, but the phenomenon is visible in any model.

Let us come back now at our resilience estimation. Considering the Arpanet model previously shown, the resilience gets pretty close to its maximal value as r gets close to 1. Some examples are given in Table 2.

For the Arpanet model used, since we have $n = 21$ nodes, the maximal possible resilience is $\binom{n}{2} = 210$. We observe that the resilience gets very close to that value, which is expected (the rare event effect). This leads to consider other ideas for differentiating situations. For instance, a way of measuring how a network reacts face to failures, is seeing what happens when there is no more total connectivity in the model. This means looking at the mean number of communicating pairs when the graph is unconnected. If we denote by *NCC* the number of connected components of our random graph G, observe first that the best situation is when there is only one component, a connected graph, and the worst is when no link is working and we have as many connected components as there are nodes, that is, $NCC = n$. So, we have $NCC \in \{1, 2, \ldots, n\}$, and on the borders, $\mathbb{P}(NCC = 1) = \mathbb{P}(NCP = \binom{n}{2}) = R_{\text{all}}$ and $\mathbb{P}(NCC = n) = \mathbb{P}(NCP = 0) = \prod_{i \in \mathcal{E}}(1 - r_i)$.

Now, a metric capturing what happens when the graph is not connected is the conditional resilience

$$\mathbb{E}(NCP \,|\, NCC \geq 2).$$

Let us illustrate this with some numerical examples. In Table 3 we show the mean number of connected components $\mathbb{E}(NCC)$, and the two resiliences just discussed.

Table 3 Three metrics on the Arpanet, i.i.c. case (homogeneous links). All the shown digits are correct (confidence level: 95%)

r	0.91	0.95	0.99	0.995	0.999
$\mathbb{E}(NCC)$	1.251	1.073	1.0027	1.00066	1.000026
$\mathbb{E}(NCP)$	199.8	207.2	209.9	209.977	209.9991
$\mathbb{E}(NCP \mid NCC \geq 2)$	160.3	167.6	174.5	175.2	175.5

The values of the conditional resilience are more separated than those of the unconditional one. But our tools allow us to make other trials. For example, take as central variable the number of pairs of nodes that can communicate using at least two edge-disjoint paths. This is related to bi-connectivity in graphs, and we will not developed the point here, but evaluating this variable needs to decompose the graph G into its bi-connected components (also a linear computation, using a DFS or a BFS, see again [10], for example). Denoting our variable as $NCP2$, we can look at the metric $\mathbb{E}(NCP2 \mid NCC \geq 2)$. We will not pursue these illustrations here. The goal is to show that evaluating these types of metrics is pretty straightforward, and only needs to be able to compute the corresponding graph analysis.

In the Appendix we provide analytical expressions to all the quantities described here in the case of a bridge, that can be useful to check algorithms and to observe behaviors.

5 Conclusions

In this chapter we first discuss the interest of moving to performability-like metrics in network reliability. This allows to distinguish several performance levels when different components fail, capturing the capacity of the system to continue to work but possibly in degraded modes. We center the chapter around one of these metrics, called resilience, that has not received yet much attention in the literature. In telecommunications, the word is being used with the same spirit but in other contexts and associated with it, there are other technologically-oriented definitions.

Then, we considered exploring these metrics with Monte Carlo. We proposed to see the standard or crude Monte Carlo procedure differently, leading to a way of implementing it with much less computational effort when components, and then systems, are highly reliable. This is just a different look at a conditional Monte Carlo approach, but it allows testing in a straightforward manner estimators for these performability metrics. Moreover, and this is how this viewpoint on standard Monte Carlo developed, we can evaluate also the sensitivities of all these metrics with respect to the (elementary) reliabilities of the systems' components, and with a very small overhead with respect to the cost of the metric evaluation itself.

In future work, we will explore appropriate ways of conditioning, depending on the metric considered, and the properties of the types of metrics we discuss here, as well as their possible uses in the area.

Appendix

For checking purposes, we put here analytical expressions of all the objects considered in Sect. 4 of the chapter, in the case of the bridge model. They are obtained basically by brute force, given the small size of the model.

- On the number of connected components NCC, $NCC \in \{1, 2, 3, 4\}$:
  - $\mathbb{P}(NCC = 1) = R_{\text{all}} = r^3(8 - 11r + 4r^2)$
  - $\mathbb{P}(NCC = 2) = 2r^2(1 - r)^2(5 - 4r) = 10r^2 - 28r^3 + 26r^4 - 8r^5$
  - $\mathbb{P}(NCC = 3) = 5r(1 - r)^4$
  - $\mathbb{P}(NCC = 4) = (1 - r)^5$
  - $\mathbb{E}(NCC) = 4 - 5r + 2r^3 + r^4 - r^5$
- On the number of communicating pairs NCP, $NCP \in \{0, 1, 2, 3, 6\}$:
  - $\mathbb{P}(NCP = 0) = (1 - r)^5$
  - $\mathbb{P}(NCP = 1) = 5r(1 - r)^4$
  - $\mathbb{P}(NCP = 2) = 2r^2(1 - r)^3$
  - $\mathbb{P}(NCP = 3) = 2r^2(1 - r)^2(4 - 3r)$
  - $\mathbb{P}(NCP = 6) = R_{\text{all}} = r^3(8 - 11r + 4r^2)$
  - $\mathbb{E}(NCP) = 5r + 8r^2 - 14r^4 + 7r^5$
  - $\mathbb{E}(NCP \,|\, NCC \geq 2) = \dfrac{r(5 + 18r - 17r^2)}{1 + 2r + 3r^2 - 4r^3}$
- On the number of pairs communicating through at least 2 edge-disjoint paths:
  - $NCP2 \in \{0, 3, 6\}$
  - $\mathbb{P}(NCP2 = 0) = (1 - r)(1 + r - r^2 + 3r^3 - 2r^4)$
  - $\mathbb{P}(NCP2 = 3) = 2r^2(1 - r)^2$
  - $\mathbb{P}(NCP2 = 6) = r^4(3 - 2r)$
  - $\mathbb{E}(NCP2) = 6r^2(1 - 2r + 4r^2 - 2r^3)$
  - $\mathbb{E}(NCP2 \,|\, NCC \geq 2) = \dfrac{6r^2}{1 + 2r + 3r^2 - 4r^3}$
- Sensitivities of the resilience:
  - W.r.t. links $\{1, 2\}$, or $\{1, 3\}$, or $\{2, 4\}$, or $\{3, 4\}$: $1 + 3r + r^2 - 12r^3 + 7r^4$;
  - w.r.t. link $\{2, 3\}$: $1 + 4r - 4r^2 - 8r^3 + 7r^4$.

References

1. Amin, A.T., Siegrist, K.T., Slater, P.J.: The expected number of pairs of connected nodes: pair-connected reliability. Mathl. Comput. Modelling **17**(11) (1993)
2. Ball, M.O., Colbourn, C.J., Provan, J.S.: Network reliability. In: Ball, M.O., Magnanti, T.L., Monma, C.L, Nemhauser, G.L. (eds.) Handbook of Operations Research: Network Models, Chap. 11, pp. 673–762. Elsevier, North-Holland, Amsterdam (1995)
3. Barrera, J., Matus, O., Moreno, E., Rubino, G.: On the Marshall-Olkin copula model for network reliability under dependent failures. IEEE Trans. Reliab. **68**(2), 451–461 (2019). https://doi.org/10.1109/TR.2018.2865707
4. Botev, Z., L'Ecuyer, P., Rubino, G., Simard, R., Tuffin, B.: Static network reliability estimation via generalized splitting. NFORMS J. Comput. **25**(1), 56–71 (2013)
5. Cancela, H., L'Ecuyer, P., Rubino, G., Tuffin, B.: Combination of conditional Monte Carlo and approximate zero-variance importance sampling for network reliability estimation. In: Proceedings of the 2010 Winter Simulation Conference, pp. 1263–1274 (2010)
6. Cancela, H., Petingi, L.: Reliability of communication networks with delay constraints: computational complexity and complete topologies. Int. J. Math. Math. Sci. **29**, 1551–1562 (2004). https://doi.org/10.1155/S016117120430623X
7. Cancela, H., Rubino, G., Tuffin, B.: New measures of robustness in rare event simulation. In: Proceedings of the 2005 Winter Simulation Conference, pp. 519–527 (2005)
8. Colbourn, C.: The Combinatorics of Network Reliability. Oxford University Press, Inc. (1987)
9. Colbourn, C.: Network resilience. SIAM J. Algebr. Discret. Methods **8**(3), 404–409 (1987). https://doi.org/10.1137/0608033
10. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 3rd edn. The MIT Press (2009)
11. El Khadiri, M., Rubino, G.: Accelerating the standard Monte Carlo evaluation of highly reliable binary systems. In: Second International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing (1996)
12. Grötshel, M., Monma, C.L., Stoer, M.: Design of survivable networks. In: Ball, M.O., Magnanti, T.L., Monma, C.L., Nemhauser, G.L. (eds.) Handbook of Operations Research: Network Models, Chap. 10, pp. 617–672. Elsevier, North-Holland, Amsterdam (1995)
13. L'Ecuyer, P., Rubino, G., Saggadi, S., Tuffin, B.: Approximate zero-variance importance sampling for static network reliability estimation. IEEE Trans. Reliab. **8**(4), 590–604 (1986)
14. Locks, M.O., Satyanarayana, A.: Network reliability—the state of the art. IEEE Trans. Reliab. **35**(3) (1986). https://doi.org/10.1109/TR.1986.4335420
15. Meyer, J.: On evaluating the performability of degradable computing systems. IEEE Trans. Comput. **C-29**(8), 720–731 (1980). https://doi.org/10.1109/TC.1980.1675654
16. Provan, J.S.: The complexity of reliability computations in planar and acyclic graphs. SIAM J. Comput. **15**(3), 694–702 (1986). https://doi.org/10.1137/0215050
17. Rubino, G.: Network reliability evaluation. In: Bagchi, K., Walrand, J. (eds.) The state-of-the art in performance modeling and simulation, Chap. 11. Gordon and Breach Books (1998)
18. Rubino, G.: Sensitivity analysis of network reliability using Monte Carlo. In: Proceedings of the 2005 Winter Simulation Conference, pp. 491–498 (2010)
19. Rubino, G., Tuffin, B. (eds.): Rare Event Simulation using Monte Carlo Methods. Wiley (2009)
20. Sebastio, S., Trivedi, K.T., Wang, D., Yin, X.: Fast computation of bounds for two-terminal network reliability. Eur. J. Oper. Res. **238**(3), 810–823 (2014). https://doi.org/10.1016/j.ejor.2014.04.03
21. Singpurwalla, N.D.: Dependence in network reliability. In: Proceedings of the Fifth International Conference on Information Fusion (FUSION 2002), vol. 2, pp. 981–985 (2002). https://doi.org/10.1109/ICIF.2002.1020918
22. Valiant, L.G.: The complexity of enumeration and reliability problems. SIAM J. Comput. **8**(3), 410–421 (1979). https://doi.org/10.1137/0208032
23. Van Slyke, R., Frank, H.: Network reliability analysis I. Networks **1**, 279–290 (1972). https://doi.org/10.1002/net.3230010307