Application of Quasi-Monte Carlo Methods to PDEs with Random Coefficients – An Overview and Tutorial

Frances Y. Kuo and Dirk Nuyens

Abstract This article provides a high-level overview of some recent works on the application of quasi-Monte Carlo (QMC) methods to PDEs with random coefficients. It is based on an in-depth survey of a similar title by the same authors, with an accompanying software package which is also briefly discussed here. Embedded in this article is a step-by-step tutorial of the required analysis for the setting known as the uniform case with first order QMC rules. The aim of this article is to provide an easy entry point for QMC experts wanting to start research in this direction and for PDE analysts and practitioners wanting to tap into contemporary QMC theory and methods.

Keywords Quasi-Monte Carlo methods · PDEs with random
coefficients · Uniform · Lognormal · Multi-level · Randomly shifted lattice rules

1 Introduction

Uncertainty quantification is the science of quantitative characterization and reduction of uncertainties in both computational and real world applications, and it is the source of many challenging high dimensional integration and approximation problems. Often the high dimensionality comes from uncertainty or randomness in the data, e.g., in groundwater flow from permeability that is rapidly varying and uncertain, or in financial mathematics from the rapid and often unpredictable changes

F. Y. Kuo (✉)
School of Mathematics and Statistics, University of New South Wales,
Sydney, NSW 2052, Australia
e-mail: f.kuo@unsw.edu.au

D. Nuyens
Department of Computer Science, KU Leuven, Celestijnenlaan 200A,
3001 Leuven, Belgium
e-mail: dirk.nuyens@cs.kuleuven.be

within markets. The input data may be a random variable or a random field, in which case the derived quantity of interest will in general also be a random variable or a random field. The computational goal is usually to find the expected value or other statistics of these derived quantities.

A popular example is the flow of water through a disordered porous medium, modeled by Darcy's law coupled with the mass conservation law, i.e.,

$$q(\boldsymbol{x}, \omega) + a(\boldsymbol{x}, \omega) \, \nabla p(\boldsymbol{x}, \omega) = 0 \,,$$
$$\nabla \cdot q(\boldsymbol{x}, \omega) = 0 \,,$$

for $\boldsymbol{x}$ in a bounded domain $D \subset \mathbb{R}^d$, $d \in \{1, 2, 3\}$, and for almost all events ω in the probability space $(\Omega, \mathscr{A}, \mathbb{P})$. Here $q(\boldsymbol{x}, \omega)$ is the velocity (also called the specific discharge) and $p(\boldsymbol{x}, \omega)$ is the residual pressure, while $a(\boldsymbol{x}, \omega)$ is the permeability (or more precisely, the ratio of permeability to dynamic viscosity) which is modelled as a random field. Uncertainty in $a(\boldsymbol{x}, \omega)$ leads to uncertainty in $q(\boldsymbol{x}, \omega)$ and $p(\boldsymbol{x}, \omega)$. Quantities of interest include for example the breakthrough time of a plume of pollution moving through the medium.

QMC for PDEs with Random Coefficients

There is a huge literature on treating these PDEs with random coefficients using various methods, see e.g., the surveys [1, 23, 43] and the references therein. Here we are interested in the application of *quasi-Monte Carlo* (*QMC*) *methods*, which are equal-weight quadrature rules for high dimensional integrals, see e.g., [3, 5, 36–39, 44].

QMC methods are still relatively new to these PDE problems. It began with the 2011 paper [21] which included comprehensive numerical experiments showing promising QMC results, but without any theoretical justification. The first fully justified theory was provided in the 2012 paper [32], and this has lead to a flood of research activities. We will follow the recent survey [31] to provide a high-level overview of how QMC theory can be applied to PDEs with random coefficients. The survey [31] covered the detailed analysis from six papers [6, 8, 22, 32–34] in a unified view. Different algorithms have been analyzed: *single-level* vs *multi-level*, *deterministic* vs *randomized*, and *first order* vs *higher order*, and they were considered under different models for the randomness as we explain below.

It is popular to assume that $a(\boldsymbol{x}, \omega)$ is a *lognormal* random field, that is, $\log(a(\boldsymbol{x}, \omega))$ is a Gaussian random field on the spatial domain D with a specified mean and covariance function. Then one can use the *Karhunen–Loève* (*KL*) *expansion* to write $\log(a(\boldsymbol{x}, \omega))$ as an infinite series parametrised by a sequence $y_j = y_j(\omega)$, $j \geq 1$, of i.i.d. standard normal random numbers from $\mathbb{R}$. Aside from the lognormal case, often the simpler *uniform* case is considered, where $a(\boldsymbol{x}, \omega)$ is written as an infinite series that depends linearly on a sequence $y_j = y_j(\omega)$, $j \geq 1$, of i.i.d. uniform random numbers from a bounded interval of $[-1, 1]$ or $[-\frac{1}{2}, \frac{1}{2}]$. In both the lognormal and uniform cases the infinite series is truncated in practice to, say, s terms. The expected value of any quantity of interest is then approximated by an s-dimensional integral with respect to the parameters y_j, which can in turn be

approximated by QMC methods, combined with finite element methods for solving the PDE.

The six papers surveyed in [31] all followed this KL-based general direction. With respect to the QMC method they can be either *first order* or *higher order*, which refers to the rate of convergence being close to $\mathcal{O}(n^{-1})$ or $\mathcal{O}(n^{-\alpha})$, $\alpha > 1$, with n being the number of integrand evaluations. With respect to the approximation of the integrand function they can be either *single-level* or *multi-level*, which refers to how spatial discretization and dimension truncation are performed. A summary of the results is given in the table below:

	Uniform case	Lognormal case
First order single-level analysis	[32]	[22]
First order multi-level analysis	[33]	[34]
Higher order single-level analysis	[6]	
Higher order multi-level analysis	[8]	

The first order results [22, 32–34] are based on *randomly shifted lattice rules* and are accompanied by probabilistic error bounds. The higher order results [6, 8] are based on *interlaced polynomial lattice rules* and are accompanied by deterministic error bounds. The lognormal results [22, 34] require a non-standard function space setting for integrands with domain $\mathbb{R}^s$. A key feature in all these analysis is that the QMC error bounds are independent of the number of integration variables s. There is as yet no satisfactory QMC theory that can give higher order convergence for the lognormal case with error bound independent of s.

Plan of this Article

In Sect. 2 we provide an overview of the different settings and algorithms covered in the survey [31], with the goal to convey the overall picture while keeping the exposition as simple and accessible as possible. In Sect. 3 we take a change of pace and style to give a step-by-step tutorial of the required analysis for the uniform case with first order QMC rules. That is, we zoom in and focus on the essence of the paper [32] in such a way that the tutorial can be used to extend the analysis to other cases by interested readers. Then in Sect. 4 we zoom out again and continue to provide insights to the key analysis required for the six papers surveyed in [31]. In Sect. 5 we briefly discuss the software accompanying [31]. Finally in Sect. 6 we give a short conclusion.

Beyond the Survey

There have been many developments beyond the scope of the survey [31].

Instead of using the KL expansion, in the lognormal case one can sample the random field only at a discrete set of points with respect to the covariance matrix inherited from the given covariance function of the continuous field. The random field is then represented exactly at these points, thus eliminating completely the truncation error associated with the KL-based approach. (Note that interpolation

may be required at the finite element quadrature nodes.) The resulting large matrix factorization problem could potentially be handled by *circulant embedding* and FFT, if the covariance function is *stationary* and the grid is regular, see [10]. In fact, this was the approach taken in the first QMC paper for PDEs with random coefficients [21], and the corresponding analysis is being considered in [19, 20].

Another way to tackle the large matrix factorization is to make use of H-*matrix techniques*, see [24], and this has been considered in [11].

The uniform framework can be extended from the elliptic PDE to the general framework of *affine* parametric operator equations, see [42] as well as [6, 8]. A different QMC theory for the lognormal case is offered in [26]. Further PDE computations with higher order QMC are reported in [14], and with multi-level and multi-index QMC in [40]. QMC has also been applied to PDEs on the sphere [35], holomorphic equations [9], Bayesian inversion [4, 41], stochastic wave propagation [12, 13], and eigenproblems [16].

Moreover, there has been some significant development in the use of functions with local support in the expansions of $a(\boldsymbol{x}, \omega)$ which leads to a simplified norm estimate for the integrand and a reduced construction cost (pre-computation) for QMC, see [15, 27, 29].

2   Overview

Throughout this article we refer to the number of integration variables s as the *stochastic dimension*, which can be in the hundreds or thousands or more (and controls the truncation error), in contrast to the *spatial dimension d* which is just 1, 2 or 3.

2.1   Uniform Versus Lognormal

For a given parameter $\boldsymbol{y}$ we consider the parametric elliptic Dirichlet problem

$$-\nabla \cdot (a(\boldsymbol{x}, \boldsymbol{y}) \nabla u(\boldsymbol{x}, \boldsymbol{y})) = \kappa(\boldsymbol{x}) \quad \text{for } \boldsymbol{x} \text{ in } D, \quad u(\boldsymbol{x}, \boldsymbol{y}) = 0 \quad \text{for } \boldsymbol{x} \text{ on } \partial D,$$
(1)

for domain $D \subset \mathbb{R}^d$ a bounded, convex, Lipschitz polyhedron with boundary ∂D, where the spatial dimension $d = 1, 2$, or 3 is assumed given and fixed. The differential operators in (1) are understood to be with respect to the physical variable $\boldsymbol{x}$ which belongs to D. The parametric variable $\boldsymbol{y} = (y_j)_{j \geq 1}$ belongs to either a bounded or unbounded domain, depending on which of the two popular formulations of the parametric coefficient $a(\boldsymbol{x}, \boldsymbol{y})$ is being considered.

Uniform Case

In the uniform case, we assume that the y_j are independent and uniformly distributed on $[-\frac{1}{2}, \frac{1}{2}]$, and

$$a(\boldsymbol{x}, \boldsymbol{y}) = a_0(\boldsymbol{x}) + \sum_{j \geq 1} y_j \, \psi_j(\boldsymbol{x}), \tag{2}$$

with $0 < a_{\min} \leq a(\boldsymbol{x}, \boldsymbol{y}) \leq a_{\max} < \infty$ for all $\boldsymbol{x}$ and $\boldsymbol{y}$. We need further assumptions on a_0 and ψ_j, see [31] for details. Here we mention only one important assumption that there exists $p_0 \in (0, 1)$ such that

$$\sum_{j \geq 1} \|\psi_j\|_{L_\infty}^{p_0} < \infty. \tag{3}$$

The value of p_0 reflects the rate of decay of the fluctuations in (2); later we will see that it directly affects the QMC convergence rate.

Our goal is to compute the integral, i.e., the expected value, with respect to $\boldsymbol{y}$, of a bounded linear functional G applied to the solution $u(\cdot, \boldsymbol{y})$ of the PDE (1)

$$\int_{\left[-\frac{1}{2}, \frac{1}{2}\right]^{\mathbb{N}}} G(u(\cdot, \boldsymbol{y})) \, \mathrm{d}\boldsymbol{y} := \lim_{s \to \infty} \int_{\left[-\frac{1}{2}, \frac{1}{2}\right]^s} G(u(\cdot, (y_1, \ldots, y_s, 0, 0, \ldots))) \, \mathrm{d}y_1 \cdots \mathrm{d}y_s. \tag{4}$$

Lognormal Case

In the lognormal case, we assume that the y_j are independent standard normal random numbers on $\mathbb{R}$, and

$$a(\boldsymbol{x}, \boldsymbol{y}) = a_0(\boldsymbol{x}) \exp\left(\sum_{j \geq 1} y_j \, \sqrt{\mu_j} \, \xi_j(\boldsymbol{x}) \right), \tag{5}$$

where $a_0(\boldsymbol{x}) > 0$, the $\mu_j > 0$ are non-increasing, and the ξ_j are orthonormal in $L_2(D)$. This can arise from the KL expansion in the case where $\log(a)$ is a stationary Gaussian random field with a specified mean and covariance function; a popular choice is the *Matérn* covariance.

Our goal now is the integral of $G(u(\cdot, \boldsymbol{y}))$ over $\boldsymbol{y} \in \mathbb{R}^{\mathbb{N}}$ with a countable product Gaussian measure $\mu_G(\mathrm{d}\boldsymbol{y})$ (formally, we restrict the domain to some $Y \subset \mathbb{R}^{\mathbb{N}}$ with full measure $\mu_G(Y) = 1$, but we omit this in the notation)

$$\int_{\mathbb{R}^{\mathbb{N}}} G(u(\cdot, \boldsymbol{y})) \prod_{j \geq 1} \phi_{\mathrm{nor}}(y_j) \, \mathrm{d}\boldsymbol{y} = \int_{[0,1]^{\mathbb{N}}} G(u(\cdot, \Phi_{\mathrm{nor}}^{-1}(\boldsymbol{w}))) \, \mathrm{d}\boldsymbol{w}, \tag{6}$$

where $\phi_{\mathrm{nor}}(y) := \exp(-y^2/2)/\sqrt{2\pi}$ is the univariate standard normal probability density function, while Φ_{nor}^{-1} denotes the inverse of the corresponding cumulative distribution function, and is applied component-wise to a vector. The transformed integral over the unit cube on the right-hand side of (6) is obtained by the change of variables $y = \Phi_{\mathrm{nor}}^{-1}(w)$.

2.2 Single-Level Versus Multi-level

Single-Level Algorithms

We approximate the integral (4) or (6) in three steps:

   i. Dimension truncation: the infinite sum in (2) or (5) is truncated to s terms.
  ii. Finite element discretization: the PDE (1) in weak formulation (see (13) below) is solved using a finite element method with meshwidth h.
 iii. QMC quadrature: the integral of the finite element solution for the truncated problem is estimated using a deterministic or randomized QMC method.

The deterministic version of this algorithm is

$$\frac{1}{n} \sum_{i=1}^{n} G(u_h^s(\cdot, y_i)), \qquad y_i = \begin{cases} t_i - \frac{1}{2} & \text{for uniform,} \\ \Phi_{\mathrm{nor}}^{-1}(t_i) & \text{for lognormal,} \end{cases} \qquad (7)$$

where $t_1, \ldots, t_n \in [0, 1]^s$ are n QMC points from the s-dimensional standard unit cube. In the uniform case, these points are translated to the unit cube $[-\frac{1}{2}, \frac{1}{2}]^s$. In the lognormal case, these points are mapped to the Euclidean space $\mathbb{R}^s$ by applying the inverse of the cumulative normal distribution function component-wise.

A randomized version of this algorithm with *random shifting* is given by

$$\frac{1}{r} \sum_{k=1}^{r} \frac{1}{n} \sum_{i=1}^{n} G(u_h^s(\cdot, y_{i,k})), \qquad y_{i,k} = \begin{cases} \{t_i + \Delta_k\} - \frac{1}{2} & \text{for uniform,} \\ \Phi_{\mathrm{nor}}^{-1}(\{t_i + \Delta_k\}) & \text{for lognormal,} \end{cases} \qquad (8)$$

where $t_1, \ldots, t_n \in [0, 1]^s$ are n QMC points as above, and $\Delta_1, \ldots, \Delta_r \in [0, 1]^s$ are r independent *random shifts* generated from the uniform distribution on $[0, 1]^s$. The braces in $\{t_i + \Delta_k\}$ mean that we take the fractional part of each component in the vector $t_i + \Delta_k$. Other randomization strategies can be used analogously but need to be chosen appropriately to preserve the special properties of the QMC points. Randomized algorithms have the advantages of being unbiased as well as providing a practical error estimate.

Multi-level Algorithms

The general concept of *multi-level* can be explained as follows (see e.g., [17]): if we denote the integral (4) or (6) by I_∞ and define a sequence $I_0, I_1, \ldots$ of

approximations converging to I_∞, then we can write I_∞ as a telescoping sum $I_\infty = (I_\infty - I_L) + \sum_{\ell=0}^{L}(I_\ell - I_{\ell-1})$, $I_{-1} := 0$, and then apply different quadrature rules to the differences $I_\ell - I_{\ell-1}$, which we anticipate to get smaller as ℓ increases. Here we define I_ℓ to be the integral of $G(u_{h_\ell}^{s_\ell})$ corresponding to the finite element solution with meshwidth h_ℓ, for the truncated problem with s_ℓ terms, where $1 \le s_0 \le s_1 \le s_2 \le \cdots \le s_L \le \cdots$ and $h_0 \ge h_1 \ge h_2 \ge \cdots \ge h_L \ge \cdots > 0$, so that I_ℓ becomes a better approximation to I_∞ as ℓ increases.

The deterministic version of our multi-level algorithm takes the form (remembering the linearity of G)

$$\sum_{\ell=0}^{L} \left(\frac{1}{n_\ell} \sum_{i=1}^{n_\ell} G((u_{h_\ell}^{s_\ell} - u_{h_{\ell-1}}^{s_{\ell-1}})(\cdot, \boldsymbol{y}_i^\ell)) \right), \qquad \boldsymbol{y}_i^\ell = \begin{cases} \boldsymbol{t}_i^\ell - \frac{1}{2} & \text{for uniform,} \\ \Phi_{\text{nor}}^{-1}(\boldsymbol{t}_i^\ell) & \text{for lognormal,} \end{cases} \tag{9}$$

where we apply an s_ℓ-dimensional QMC rule with n_ℓ points $\boldsymbol{t}_1^\ell, \ldots, \boldsymbol{t}_{n_\ell}^\ell \in [0, 1]^{s_\ell}$ to the integrand $G(u_{h_\ell}^{s_\ell} - u_{h_{\ell-1}}^{s_{\ell-1}})$, and we define $u_{h_{-1}}^{s_{-1}} := 0$.

The corresponding randomized version can be obtained analogously to (8) by taking r_ℓ random shifts at each level, noting that all shifts from all levels should be independent.

2.3   First-Order Versus Higher-Order

Up to this point we have said very little about QMC methods, other than noting that they are equal-weight quadrature rules as seen in (7). Actually, we will not say much about QMC methods in this article at all. In this subsection we will mention three different QMC theoretical settings which have been used for PDEs applications, giving just enough details in the first setting needed for the tutorial in Sect. 3. These three settings are discussed in slightly more detail in [30] in this volume, and more comprehensively in [31]; see also the references in these papers.

First Order QMC Over the Unit Cube – Randomly Shifted Lattice Rules for Weighted Sobolev Spaces

Suppose we wish to approximate the s-dimensional integral over the unit cube $[0, 1]^s$

$$\int_{[0,1]^s} f(\boldsymbol{y}) \, d\boldsymbol{y} \,, \tag{10}$$

where the integrand f belongs to a *weighted Sobolev space of smoothness one*, with the *unanchored* norm defined by (see e.g., [45])

$$\|f\|_\gamma = \left[\sum_{\mathfrak{u} \subseteq \{1:s\}} \frac{1}{\gamma_\mathfrak{u}} \int_{[0,1]^{|\mathfrak{u}|}} \left(\int_{[0,1]^{s-|\mathfrak{u}|}} \frac{\partial^{|\mathfrak{u}|} f}{\partial \boldsymbol{y}_\mathfrak{u}}(\boldsymbol{y}) \, d\boldsymbol{y}_{\{1:s\}\setminus\mathfrak{u}} \right)^2 d\boldsymbol{y}_\mathfrak{u} \right]^{1/2} . \tag{11}$$

Here $\{1 : s\}$ is a shorthand notation for the set of indices $\{1, 2, \ldots, s\}$, $(\partial^{|\mathfrak{u}|} f)/(\partial \boldsymbol{y}_{\mathfrak{u}})$ denotes the mixed first derivative of f with respect to the "active" variables $\boldsymbol{y}_{\mathfrak{u}} = (y_j)_{j \in \mathfrak{u}}$, while $\boldsymbol{y}_{\{1:s\} \setminus \mathfrak{u}} = (y_j)_{j \in \{1:s\} \setminus \mathfrak{u}}$ denotes the "inactive" variables. There is a weight parameter $\gamma_{\mathfrak{u}} \geq 0$ associated with each subset of variables $\boldsymbol{y}_{\mathfrak{u}}$ to moderate the relative importance between the different sets of variables. We denote the weights collectively by $\boldsymbol{\gamma}$.

In this setting we pair the weighted Sobolev space with *randomly shifted lattice rules*; the complete theory can be found in [5]. They approximate the integral (10) by

$$\frac{1}{n} \sum_{i=1}^{n} f(\boldsymbol{t}_i), \qquad \boldsymbol{t}_i = \left\{ \frac{i\,\boldsymbol{z}}{n} + \boldsymbol{\Delta} \right\},$$

where $\boldsymbol{z} \in \mathbb{Z}^s$ is known as the *generating vector*, $\boldsymbol{\Delta}$ is a *random shift* drawn from the uniform distribution over $[0, 1]^s$, and as in (8) the braces indicate that we take the fractional parts of each component in a vector. It is known that good generating vectors can be obtained using a *CBC construction* (*component-by-component construction*), determining the components of $\boldsymbol{z}$ one at a time sequentially, to achieve first order convergence in this setting, where the implied constant can be independent of s under appropriate conditions on the weights $\boldsymbol{\gamma}$.

Specifically, if n is a power of 2 then we know that the CBC construction yields the root-mean-square error bound (with respect to the uniform random shift), for all $\lambda \in (1/2, 1]$,

$$\text{r.m.s. error} \leq \left(\frac{2}{n} \sum_{\emptyset \neq \mathfrak{u} \subseteq \{1:s\}} \gamma_{\mathfrak{u}}^{\lambda} \left[\vartheta(\lambda) \right]^{|\mathfrak{u}|} \right)^{1/(2\lambda)} \| f \|_{\boldsymbol{\gamma}}, \qquad (12)$$

where $\vartheta(\lambda) := 2\zeta(2\lambda)/(2\pi^2)^{\lambda}$, with $\zeta(a) := \sum_{k=1}^{\infty} k^{-a}$ denoting the Riemann zeta function. A similar result holds for general n. The best rate of convergence clearly comes from choosing λ close to $1/2$.

We need some structure in the weights $\boldsymbol{\gamma}$ for the CBC construction cost to be feasible in practice. *Fast* CBC algorithms (using FFT) can find a generating vector of a good n-point lattice rule in s dimensions in $\mathcal{O}(s\,n \log n)$ operations in the case of *product weights*, and in $\mathcal{O}(s\,n \log n + s^2\,n)$ operations in the case of *POD weights* (see (25) ahead).

First Order QMC Over $\mathbb{R}^s$

We can pair randomly shifted lattice rules with a special function space setting over $\mathbb{R}^s$ to achieve first order convergence. The norm in this function space setting includes some additional weight functions to control the behavior of the derivatives of the functions as the components go to $\pm\infty$. The root-mean-square error bound takes the same form as (12), but with a different definition of the norm and $\vartheta(\lambda)$.

Higher Order QMC Over the Unit Cube

We can pair a family of QMC methods called *interlaced polynomial lattice rules* with another special function space setting over the unit cube to achieve higher order convergence. The norm in this function space setting involves higher order mixed derivatives of the functions. The key advantage of this family of QMC methods over other higher order QMC methods is that, in the cost of finding a generating vector which achieves the best theoretical convergence rate, the *order* or the *interlacing factor* appears as a multiplying factor rather than sitting in the exponent of the number of points n.

3 Tutorial

We conclude from the error bound (12) that the first step in applying QMC theory is to estimate the norm of the practical integrand. We see from (7), (8), and (9) that this means we need to estimate the norms

$$\|G(u_h^s)\|_{\gamma} \quad \text{and} \quad \|G(u_{h_\ell}^{s_\ell} - u_{h_{\ell-1}}^{s_{\ell-1}})\|_{\gamma},$$

for the single-level and the multi-level algorithms, respectively.

In this section we provide a step-by-step tutorial on the analysis for the single-level algorithm in the uniform case with first order QMC rules.

Differentiate the PDE

1. We start with the variational formulation of the PDE (1): find $u(\cdot, y) \in H_0^1(D)$ such that

$$\int_D a(x, y) \nabla u(x, y) \cdot \nabla w(x) \, dx = \int_D \kappa(x) \, w(x) \, dx \quad \forall w \in H_0^1(D). \quad (13)$$

   Here we consider the Sobolev space $H_0^1(D)$ of functions which vanish on the boundary of D, with norm $\|w\|_{H_0^1} := \|\nabla w\|_{L_2}$, and together with the dual space $H^{-1}(D)$ and pivot space $L_2(D)$.

2. We take the mixed partial derivatives ∂^{ν} with respect to y with multi-index $\nu \neq \mathbf{0}$ (i.e., we differentiate ν_j times with respect to y_j for each j) on both sides of (13) to obtain

$$\int_D \partial^{\nu}\Big(a(x, y) \nabla u(x, y) \cdot \nabla w(x)\Big) \, dx = 0 \quad \forall w \in H_0^1(D). \quad (14)$$

   We can move the derivatives inside the integrals because they operate on different variables y and x, respectively. The right-hand side vanishes because it does not depend on y.

3. Next we apply the Leibniz product rule on the left-hand side of (14) to obtain

$$\int_D \left(\sum_{\boldsymbol{m} \le \boldsymbol{v}} \binom{\boldsymbol{v}}{\boldsymbol{m}} (\partial^{\boldsymbol{m}} a)(\boldsymbol{x}, \boldsymbol{y}) \, \nabla(\partial^{\boldsymbol{v}-\boldsymbol{m}} u)(\boldsymbol{x}, \boldsymbol{y}) \cdot \nabla w(\boldsymbol{x}) \right) \mathrm{d}\boldsymbol{x} = 0 \quad \forall w \in H_0^1(D), \quad (15)$$

where the sum is over all multi-indices $\boldsymbol{m}$ satisfying $\boldsymbol{m} \le \boldsymbol{v}$ (i.e., $m_j \le v_j$ for all j), and $\binom{\boldsymbol{v}}{\boldsymbol{m}} := \prod_{j \ge 1} \binom{v_j}{m_j}$. So far we have made no use of any assumption on $a(\boldsymbol{x}, \boldsymbol{y})$.

4. For the uniform case, it is easy to see from the formula (2) of $a(\boldsymbol{x}, \boldsymbol{y})$ that

$$(\partial^{\boldsymbol{m}} a)(\boldsymbol{x}, \boldsymbol{y}) = \begin{cases} a(\boldsymbol{x}, \boldsymbol{y}) & \text{if } \boldsymbol{m} = \boldsymbol{0}, \\ \psi_j(\boldsymbol{x}) & \text{if } \boldsymbol{m} = \boldsymbol{e}_j, \\ 0 & \text{otherwise}, \end{cases} \quad (16)$$

where $\boldsymbol{e}_j$ denotes the multi-index whose jth component is 1 and all other components are 0. Essentially, due to the linearity of a with respect to each y_j, if we differentiate once then we obtain ψ_j, and if we differentiate a second time with respect to any variable we get 0.

5. Substituting (16) into (15) and separating out the $\boldsymbol{m} = \boldsymbol{0}$ term, we obtain

$$\int_D a(\boldsymbol{x}, \boldsymbol{y}) \, \nabla(\partial^{\boldsymbol{v}} u)(\boldsymbol{x}, \boldsymbol{y}) \cdot \nabla w(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}$$
$$= -\sum_{j \ge 1} v_j \int_D \psi_j(\boldsymbol{x}) \, \nabla(\partial^{\boldsymbol{v}-\boldsymbol{e}_j} u)(\boldsymbol{x}, \boldsymbol{y}) \cdot \nabla w(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \quad \forall w \in H_0^1(D). \quad (17)$$

6. Note that (17) holds for all test functions in $H_0^1(D)$. We now take the particular choice of $w = (\partial^{\boldsymbol{v}} u)(\cdot, \boldsymbol{y})$ (yes, it is allowed to depend on $\boldsymbol{y}$) in (17). Applying $a(\boldsymbol{x}, \boldsymbol{y}) \ge a_{\min}$ to the left-hand side, and $|\psi_j(\boldsymbol{x})| \le \|\psi_j\|_{L_\infty}$ and the Cauchy–Schwarz inequality to the right-hand side, we obtain

$$a_{\min} \int_D |\nabla(\partial^{\boldsymbol{v}} u)(\boldsymbol{x}, \boldsymbol{y})|^2 \, \mathrm{d}\boldsymbol{x} \quad (18)$$
$$\le \sum_{j \ge 1} v_j \, \|\psi_j\|_{L_\infty} \left(\int_D |\nabla(\partial^{\boldsymbol{v}-\boldsymbol{e}_j} u)(\boldsymbol{x}, \boldsymbol{y})|^2 \, \mathrm{d}\boldsymbol{x} \right)^{1/2} \left(\int_D |\nabla(\partial^{\boldsymbol{v}} u)(\boldsymbol{x}, \boldsymbol{y})|^2 \, \mathrm{d}\boldsymbol{x} \right)^{1/2}.$$

7. Canceling one common factor from both sides of (18) and then dividing through by $a_{\min}$, we obtain the recurrence

$$\|\nabla(\partial^{\boldsymbol{v}} u)(\cdot, \boldsymbol{y})\|_{L_2} \le \sum_{j \ge 1} v_j \, b_j \, \|\nabla(\partial^{\boldsymbol{v}-\boldsymbol{e}_j} u)(\cdot, \boldsymbol{y})\|_{L_2}, \qquad b_j := \frac{\|\psi_j\|_{L_\infty}}{a_{\min}}.$$
$$(19)$$

8. Finally we prove by induction that

$$\|\nabla(\partial^{\boldsymbol{v}}u)(\cdot,\boldsymbol{y})\|_{L_2} \leq |\boldsymbol{v}|!\,\boldsymbol{b}^{\boldsymbol{v}}\,\frac{\|\kappa\|_{H^{-1}}}{a_{\min}}\,, \tag{20}$$

where $|\boldsymbol{v}| := \sum_{j\geq 1} v_j$ and $\boldsymbol{b}^{\boldsymbol{v}} := \prod_{j\geq 1} b_j^{v_j}$.

a. BASE STEP. We return to the variational form (13) and take $w = u(\cdot,\boldsymbol{y})$. Applying $a(\boldsymbol{x},\boldsymbol{y}) \geq a_{\min}$ to the left-hand side and estimating the right-hand side using duality pairing $|\langle\kappa, u(\cdot,\boldsymbol{y})\rangle| \leq \|\kappa\|_{H^{-1}} \|u(\cdot,\boldsymbol{y})\|_{H_0^1}$, we obtain

$$a_{\min} \|\nabla u(\cdot,\boldsymbol{y})\|_{L_2}^2 \leq \|\kappa\|_{H^{-1}} \|\nabla u(\cdot,\boldsymbol{y})\|_{L_2}\,,$$

which can be rearranged to yield the case $\boldsymbol{v} = \boldsymbol{0}$ in (20).

b. INDUCTION STEP. As the induction hypothesis, we assume that (20) holds for all multi-indices of order $< |\boldsymbol{v}|$. Then we have

$$\|\nabla(\partial^{\boldsymbol{v}-\boldsymbol{e}_j}u)(\cdot,\boldsymbol{y})\|_{L_2} \leq |\boldsymbol{v}-\boldsymbol{e}_j|!\,\boldsymbol{b}^{\boldsymbol{v}-\boldsymbol{e}_j}\,\frac{\|\kappa\|_{H^{-1}}}{a_{\min}}\,.$$

Substituting this into (19) and noting that $v_j\,|\boldsymbol{v}-\boldsymbol{e}_j|! = |\boldsymbol{v}|!$ and $b_j\,\boldsymbol{b}^{\boldsymbol{v}-\boldsymbol{e}_j} = \boldsymbol{b}^{\boldsymbol{v}}$, we obtain (20) and conclude the induction.

Estimate the Norm

9. We want to estimate the norm $\|G(u_h^s)\|_{\boldsymbol{\gamma}}$. We see from the definition of the norm in (11) that we need to obtain estimates on the mixed first derivatives of $G(u_h^s(\cdot,\boldsymbol{y}))$ with respect to $\boldsymbol{y}$. Using linearity and boundedness of G, we have

$$\left|\frac{\partial^{|\mathfrak{u}|}}{\partial\boldsymbol{y}_{\mathfrak{u}}}G(u_h^s(\cdot,\boldsymbol{y}))\right| = \left|G\left(\frac{\partial^{|\mathfrak{u}|}}{\partial\boldsymbol{y}_{\mathfrak{u}}}u_h^s(\cdot,\boldsymbol{y})\right)\right| \leq \|G\|_{H^{-1}}\left\|\frac{\partial^{|\mathfrak{u}|}}{\partial\boldsymbol{y}_{\mathfrak{u}}}u_h^s(\cdot,\boldsymbol{y})\right\|_{H_0^1}. \tag{21}$$

10. We can repeat the above proof of (20) for the truncated finite element solution u_h^s instead of the true solution u. Then we restrict the result to mixed first derivatives (i.e., $v_j \leq 1$ for all j) and deduce that

$$\left\|\frac{\partial^{|\mathfrak{u}|}}{\partial\boldsymbol{y}_{\mathfrak{u}}}u_h^s(\cdot,\boldsymbol{y})\right\|_{H_0^1} \leq |\mathfrak{u}|!\left(\prod_{j\in\mathfrak{u}}b_j\right)\frac{\|\kappa\|_{H^{-1}}}{a_{\min}}, \quad \mathfrak{u}\subseteq\{1:s\}\,. \tag{22}$$

11. Combining (21) with (22) and substituting the upper bound into the definition of the norm (11), we conclude that

$$\|G(u_h^s)\|_{\boldsymbol{\gamma}} \leq \frac{\|\kappa\|_{H^{-1}}\|G\|_{H^{-1}}}{a_{\min}}\left(\sum_{\mathfrak{u}\subseteq\{1:s\}}\frac{(|\mathfrak{u}|!)^2\prod_{j\in\mathfrak{u}}b_j^2}{\gamma_{\mathfrak{u}}}\right)^{1/2}. \tag{23}$$

Choose the Weights

12. Now we apply the upper bound on the norm (23) in the error bound for randomly shifted lattice rules (12), to yield (leaving out some constants as indicated by $\lesssim$) for all $\lambda \in (1/2, 1]$,

$$\text{r.m.s. error} \lesssim \left(\frac{2}{n} \sum_{\mathfrak{u} \subseteq \{1:s\}} \gamma_{\mathfrak{u}}^{\lambda} \left[\vartheta(\lambda) \right]^{|\mathfrak{u}|} \right)^{1/(2\lambda)} \left(\sum_{\mathfrak{u} \subseteq \{1:s\}} \frac{(|\mathfrak{u}|!)^2 \prod_{j \in \mathfrak{u}} b_j^2}{\gamma_{\mathfrak{u}}} \right)^{1/2}. \tag{24}$$

13. With elementary calculus, for any λ, we can minimize the the upper bound in (24) with respect to the weights $\gamma_{\mathfrak{u}}$ to yield the formula

$$\gamma_{\mathfrak{u}} = \left(|\mathfrak{u}|! \prod_{j \in \mathfrak{u}} \frac{b_j}{\sqrt{\vartheta(\lambda)}} \right)^{2/(1+\lambda)}. \tag{25}$$

This form of weights is called *product and order dependent weights*, or *POD weights* in short, because of the presence of some product factors as well as the cardinality of $\mathfrak{u}$.

14. We substitute (25) into (24) and simplify the expression to

$$\text{r.m.s. error} \tag{26}$$
$$\lesssim \left(\frac{2}{n} \right)^{1/(2\lambda)} \left[\sum_{\mathfrak{u} \subseteq \{1:s\}} \left(|\mathfrak{u}|! \prod_{j \in \mathfrak{u}} \left(b_j \left[\vartheta(\lambda) \right]^{1/(2\lambda)} \right) \right)^{2\lambda/(1+\lambda)} \right]^{(1+\lambda)/(2\lambda)}.$$

15. We now derive a condition on λ for which the sum in (26) is bounded independently of s. In an abstract form, we have

$$\sum_{\mathfrak{u} \subseteq \{1:s\}} \left(|\mathfrak{u}|! \prod_{j \in \mathfrak{u}} \alpha_j \right)^k = \sum_{\ell=0}^{s} (\ell!)^k \sum_{\mathfrak{u} \subseteq \{1:s\}, |\mathfrak{u}|=\ell} \prod_{j \in \mathfrak{u}} \alpha_j^k \leq \sum_{\ell=0}^{s} (\ell!)^{k-1} \left(\sum_{j=1}^{s} \alpha_j^k \right)^{\ell},$$

where the inequality holds because each term $\prod_{j \in \mathfrak{u}} \alpha_j^k$ from the left-hand side of the inequality appears in the expansion $\left(\sum_{j=1}^{s} \alpha_j^k \right)^{\ell}$ exactly $\ell!$ times and yet the expansion contains other terms. The right-hand side is bounded independently of s if $\sum_{j=1}^{\infty} \alpha_j^k < \infty$ and $k < 1$, which can be verified by the ratio test. In our case, we have $k = 2\lambda/(1 + \lambda)$ and $\sum_{j=1}^{\infty} \alpha_j^k = [\vartheta(\lambda)]^{1/(1+\lambda)} \sum_{j=1}^{\infty} b_j^k < \infty$ if $k \geq p_0$, where we recall that b_j is defined in (19) and p_0 is defined in (3). Hence we require

$$p_0 \leq \frac{2\lambda}{1 + \lambda} < 1 \iff \frac{p_0}{2 - p_0} \leq \lambda < 1. \tag{27}$$

16. Clearly the best rate of convergence is obtained by taking λ as small as possible. Combining the original constraint of $\lambda \in (1/2, 1]$ with (27), we now take

$$
\lambda = \begin{cases} \dfrac{1}{2 - 2\delta} & \text{for } \delta \in \left(0, \frac{1}{2}\right) & \text{when } p_0 \in \left(0, \frac{2}{3}\right], \\[3mm] \dfrac{p_0}{2 - p_0} & & \text{when } p_0 \in \left(\frac{2}{3}, 1\right). \end{cases} \tag{28}
$$

Fast CBC Construction

17. The chosen weights (25) with λ given by (28) are then fed into the CBC construction to produce tailored randomly shifted lattice rules that achieve a root-mean-square error of order

$$
n^{-\min(1/p_0 - 1/2, 1 - \delta)}, \quad \delta \in \left(0, \frac{1}{2}\right),
$$

with the implied constant independent of s, where p_0 is given by (3). The fast CBC construction with POD weights can then find a good generating vector in $\mathcal{O}(s\, n \log n + s^2\, n)$ operations.

4 Key Analysis

Having completed our embedded tutorial in the previous section, we now continue to provide our overview of the analysis required in applying QMC to PDEs with random coefficients.

Some Hints at the Technical Difficulties for the Multi-level Analysis

We have seen in the uniform case with the single-level algorithm that the key is to estimate $\|G(u_h^s)\|_{\boldsymbol{\gamma}}$, and this is achieved by estimating (see (20) and [31, Lemma 6.1])

$$
\|\nabla \partial^{\boldsymbol{\nu}} u(\cdot, \boldsymbol{y})\|_{L_2}.
$$

For the multi-level algorithm, the key estimate is $\|G(u_{h_\ell}^{s_\ell} - u_{h_{\ell-1}}^{s_{\ell-1}})\|_{\boldsymbol{\gamma}}$, and we need to estimate in turn (see [31, Lemmas 6.2–6.4])

$$
\|\Delta \partial^{\boldsymbol{\nu}} u(\cdot, \boldsymbol{y})\|_{L_2}, \quad \|\nabla \partial^{\boldsymbol{\nu}} (u - u_h)(\cdot, \boldsymbol{y})\|_{L_2}, \quad \text{and} \quad |\partial^{\boldsymbol{\nu}} G((u - u_h)(\cdot, \boldsymbol{y}))|.
$$

All three bounds involve factors of the form $(|\boldsymbol{\nu}| + a_1)!\, \overline{\boldsymbol{b}}^{\boldsymbol{\nu}}$ for $a_1 \geq 0$ and a sequence $\overline{b}_j$ similar to the previously defined b_j. Assuming that both the forcing term κ and the linear functional G are in $L_2(D)$, we obtain that the second bound is of order h and the third bound is of order h^2. The difficulty is that we need to establish these regularity estimates simultaneously in $\boldsymbol{x}$ and $\boldsymbol{y}$. We also use duality tricks to gain on the convergence rate due to the linear functional G.

Some Hints at the Technical Difficulties for the Lognormal Case

For the lognormal case the argument is quite technical due to the more complicated form of $a(x, y)$. In the single-level algorithm we need to estimate (see [31, Lemma 6.5])

$$\|\nabla \partial^\nu u(\cdot, y)\|_{L_2} \quad \text{by first estimating} \quad \|a^{1/2}(\cdot, y) \nabla \partial^\nu u(\cdot, y)\|_{L_2}.$$

In the multi-level algorithm we need to estimate (see [31, Lemma 6.6])

$$\|\Delta \partial^\nu u(\cdot, y)\|_{L_2} \quad \text{by first estimating} \quad \|a^{-1/2}(\cdot, y) \nabla \cdot (a(\cdot, y) \nabla \partial^\nu u(\cdot, y))\|_{L_2},$$

and then estimate in turn (see [31, Lemmas 6.7–6.8])

$$\|a^{1/2}(\cdot, y) \nabla \partial^\nu (u - u_h)(\cdot, y)\|_{L_2} \quad \text{and} \quad |\partial^\nu G((u - u_h)(\cdot, y))|.$$

All bounds involve factors of the form $J(y) (|\nu| + a_1)! \, \beta^\nu$ for $a_1 \geq 0$ and some sequence β_j, where $J(y)$ indicates some factor depending on y which is not present in the uniform case. The proofs are by induction, and the tricky part is knowing what multiplying factor of $a(\cdot, y)$ should be included in the recursion. The growth of $J(y)$ needs to be taken into account when estimating the norm.

Summary of Results

Now we summarize and compare the results from [6, 8, 32, 33] for the uniform case:

First-order single-level [33]
$$s^{-2(1/p_0 - 1)} + h^{t+t'} + n^{-\min(1/p_0 - 1/2, 1 - \delta)} \quad \text{(r.m.s.)}$$

First-order multi-level [34]
$$s_L^{-2(1/p_0 - 1)} + h_L^{t+t'} + \sum_{\ell=0}^{L} n_\ell^{-\min(1/p_1 - 1/2, 1 - \delta)} \left(\theta_{\ell-1} \, s_{\ell-1}^{-(1/p_0 - 1/p_1)} + h_{\ell-1}^{t+t'} \right) \quad \text{(r.m.s.)}$$

Higher-order single-level [4]
$$s^{-2(1/p_0 - 1)} + h^{t+t'} + n^{-1/p_0}$$

Higher-order multi-level [6]
$$s_L^{-2(1/p_0 - 1)} + h_L^{t+t'} + \sum_{\ell=0}^{L} n_\ell^{-1/p_t} \left(\theta_{\ell-1} \, s_{\ell-1}^{-(1/p_0 - 1/p_t)} + h_{\ell-1}^{t+t'} \right)$$

For the first-order results, the "r.m.s." in brackets indicates that the error is in the root-mean-square sense since we use a randomized QMC method. The higher-order results are deterministic. Without giving the full details, we simply say that the results include general parameters t and t' for the regularity of κ and G, respectively. Recall that p_0 corresponds to the summability of $\|\psi_j\|_{L_\infty}$, see (3). Here p_1 corresponds essentially to the summability of $\|\nabla \psi_j\|_{L_\infty}$, while p_t corresponds analogously to

higher derivatives of ψ_j. For the multi-level results we include the analysis for potentially taking different s_ℓ at each level: $\theta_{\ell-1}$ is 0 if $s_\ell = s_{\ell-1}$ and is 1 otherwise.

In the single-level algorithms, the error is the sum of three terms. In the multi-level algorithms, we see the multiplicative effect between the finite element error and the QMC error. However, comparing p_1 and p_t with p_0, we see that multi-level algorithms need stronger regularity in x than single-level algorithms.

Going from first-order to higher-order results, we see that the cap of $n^{-(1-\delta)}$ is removed. We also see a gain of an extra factor of $n^{-1/2}$; this benefit appears to arise from the switch of function space setting to a non-Hilbert space.

The error versus cost analysis depends crucially on the cost assumptions. For the single-level algorithms, we simply choose n, s and h to balance three errors. In the multi-level algorithms, we choose n_ℓ, s_ℓ, h_ℓ to minimize the total cost for a fixed total error using Lagrange multiplier arguments.

For the lognormal case we have similar first order results, see [22, 34]. There is no higher order results for the lognormal case because presently there is no QMC theory in this setting.

5 Software

The software package QMC4PDE accompanies the survey [31], see https://people. cs.kuleuven.be/~dirk.nuyens/qmc4pde/. Here we very briefly outline its usage.

Construction of the Generating Vector in Python

In the analysis for the PDE problems we obtain generic bounds on mixed derivatives of the form

$$|\partial^{\nu} F(y)| \lesssim \big((|\nu| + a_1)!\big)^{d_1} \prod_{j=1}^{s} (a_2 \mathscr{B}_j)^{\nu_j} \exp(a_3 \mathscr{B}_j |y_j|),$$

for some constants a_1, a_2, a_3, d_1 and some sequence $\mathscr{B}_j$, where

$$F(y) = \begin{cases} G(u_h^s) & \text{for single-level algorithms,} \\ G(u_{h_\ell}^s - u_{h_{\ell-1}}^s) & \text{for multi-level algorithms,} \end{cases}$$

and in particular

$$\begin{cases} a_3 = 0 & \text{for the uniform case,} \\ a_3 > 0 & \text{for the lognormal case.} \end{cases}$$

The Python construction script takes the number of points (as a power of 2), the dimension, and all these parameters as input from the user, works out the appropriate weights γ_u, and then constructs a good generating vector for the QMC rule. This is either a lattice sequence (constructed following a minimax strategy as described

in [2]) or an interlaced polynomial lattice rule. In the latter case the script also assembles the interlaced generating matrices, because this is the most convenient way to generate the points.

- To construct a generating vector for a lattice sequence (output written to file `z.txt`)

```
## uniform case, 100-dim, 2^10 points, with specified bounds b:
./lat-cbc.py --s=100 --m=10 --d2=3 --b="0.1 * j**-3 / log(j+1)"
```

```
## lognormal case, 100-dim, 2^10 points, with algebraic decay:
./lat-cbc.py --s=100 --m=10 --a2="1/log(2)" --a3=1 --d2=3 --c=0.1
```

- To construct generating matrices for an interlaced polynomial lattice rule (output written to file `Bs53.col`)

```
## 100-dim, 2^10 points, interlacing 3, with bounds from file:
./polylat-cbc.py --s=100 --m=10 --alpha=3 --a1=5 --b_file=in.txt
```

Point Generators in Matlab/Octave (also available in C++ and Python)

Here are some Matlab/Octave usage examples for generating the actual QMC point sets from the output files of the Python construction script.

- To generate a lattice sequence (specified by the file `z.txt`)

```
load z.txt                        % load generating vector
latticeseq_b2('init0', z)         % initialize the generator
Pa = latticeseq_b2(20, 512);      % first 512 20-dim points
Pb = latticeseq_b2(20, 512);      % next 512 20-dim points
```

- To generate an interlaced polynomial lattice rule (specified by the file `Bs53.col`)

```
load Bs53.col                     % load generating matrices
digitalseq_b2g('init0', Bs53)     % initialize the generator
Pa = digitalseq_b2g(100, 512);    % first 512 100-dim points
Pb = digitalseq_b2g(100, 512);    % next 512 100-dim points
```

The same function `digitalseq_b2g` can also be used to generate interlaced Sobol′ points by specifying the corresponding interlaced generating matrices. The parameters for generating Sobol′ points are taken from [28].

- To generate an interlaced Sobol′ sequence (interlaced matrices specified by the file `sobol_alpha3_Bs53.col`)

```
load sobol_alpha3_Bs53.col          % load generating matrices
digitalseq_b2g('init0', sobol_alpha3_Bs53)  % initialize
Pa = digitalseq_b2g(50, 512);}      % first 512 50-dim
Pb = digitalseq_b2g(50, 512);}      % next 512 50-dim
```

The last example produces interlaced Sobol′ points with interlacing factor $\alpha = 3$. They can provide third order convergence if the integrand has sufficient smoothness.

6 Concluding Remarks

QMC (deterministic or randomized) convergence rate and implied constant can be independent of the dimension. This is achieved by working in a weighted function space setting. To apply QMC theory, we need an estimate of the norm of the integrand, and in turn this can help us to choose appropriate weights for the function space. The chosen weights then enter the fast CBC construction of the generating vector for the QMC points. The pairing between the function space setting and the QMC method is very important, in the sense that we want to achieve the best possible convergence rate under the weakest assumption on the problem. In practice, it may be that an off-the-shelf QMC rule works just as well, barring no theory.

In this article we considered multi-level algorithms. There are other cost saving strategies for the lognormal case and for other general situations, see e.g., [7, 25] as well as [18, 30] in this volume. Moreover, there have been many others developments on the application of QMC to PDEs with random coefficients, for some examples see the last part of the introduction.

References

1. Cohen, A., DeVore, R.: Approximation of high-dimensional parametric PDEs. Acta Numer. **24**, 1–159 (2015)
2. Cools, R., Kuo, F.Y., Nuyens, D.: Constructing embedded lattice rules for multivariate integration. SIAM J. Sci. Comput. **28**, 2162–2188 (2006)
3. Dick, J., Pillichshammer, F.: Digital Nets and Sequences. Cambridge University Press, Cambridge (2010)
4. Dick, J., Gantner, R.N., Le Gia, Q.T., Schwab, Ch.: Higher order Quasi-Monte Carlo integration for Bayesian estimation (in review)

 5. Dick, J., Kuo, F.Y., Sloan, I.H.: High-dimensional integration: the Quasi-Monte Carlo way. Acta Numer. **22**, 133–288 (2013)
 6. Dick, J., Kuo, F.Y., Le Gia, Q.T., Nuyens, D., Schwab, Ch.: Higher order QMC Galerkin discretization for parametric operator equations. SIAM J. Numer. Anal. **52**, 2676–2702 (2014)
 7. Dick, J., Kuo, F.Y., Le Gia, Q.T., Schwab, Ch.: Fast QMC matrix-vector multiplication. SIAM J. Sci. Comput. **37**, A1436–A1450 (2015)
 8. Dick, J., Kuo, F.Y., Le Gia, Q.T., Schwab, Ch.: Multi-level higher order QMC Galerkin discretization for affine parametric operator equations. SIAM J. Numer. Anal. **54**, 2541–2568 (2016)
 9. Dick, J., Le Gia, Q.T., Schwab, Ch.: Higher order Quasi-Monte Carlo integration for holomorphic, parametric operator equations. SIAM/ASA J. Uncertain. Quantif. **4**, 48–79 (2016)
10. Dietrich, C.R., Newsam, G.H.: Fast and exact simulation of stationary Gaussian processes through circulant embedding of the covariance matrix. SIAM J. Sci. Comput. **18**, 1088–1107 (1997)
11. Feischl, M., Kuo, F.Y., Sloan, I.H.: Fast random field generation with H-matrices. Numer. Math. (to appear)
12. Ganesh, M., Hawkins, S.C.: A high performance computing and sensitivity analysis algorithm for stochastic many-particle wave scattering. SIAM J. Sci. Comput. **37**, A1475–A1503 (2015)
13. Ganesh, M., Kuo, F.Y., Sloan, I.H.: Quasi-Monte Carlo finite element wave propagation in heterogeneous random media (in preparation)
14. Gantner, R.N., Schwab, Ch.: Computational higher order quasi-Monte Carlo integration. In: Nuyens, D., Cools, R. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2014, pp. 271–288. Springer, Berlin (2016)
15. Gantner, R.N., Herrmann, L., Schwab, Ch.: Quasi-Monte Carlo integration for affine-parametric, elliptic PDEs: local supports and product weights SIAM. J. Numer. Anal. **56**(1), 111–135 (2018)
16. Gilbert, A.D., Graham, I.G., Kuo, F.Y., Scheichl, R., Sloan, I.H.: Analysis of quasi-Monte Carlo methods for elliptic eigenvalue problems with stochastic coefficients (in preparation)
17. Giles, M.B.: Multilevel Monte Carlo methods. Acta Numer. **24**, 259–328 (2015)
18. Giles, M.B., Kuo, F.Y., Sloan, I.H.: Combining sparse grids, multilevel MC and QMC for elliptic PDEs with random coefficients (in this volume)
19. Graham, I.G., Kuo, F.Y., Nuyens, D., Scheichl, R., Sloan, I.H.: Analysis of circulant embedding methods for sampling stationary random fields. SIAM J. Numer. Anal. (to appear)
20. Graham, I.G., Kuo, F.Y., Nuyens, D., Scheichl, R., Sloan, I.H.: Circulant embedding with QMC – analysis for elliptic PDE with lognormal coefficients. Numer. Math. (to appear)
21. Graham, I.G., Kuo, F.Y., Nuyens, D., Scheichl, R., Sloan, I.H.: Quasi-Monte Carlo methods for elliptic PDEs with random coefficients and applications. J. Comput. Phys. **230**, 3668–3694 (2011)
22. Graham, I.G., Kuo, F.Y., Nichols, J.A., Scheichl, R., Schwab, Ch., Sloan, I.H.: QMC FE methods for PDEs with log-normal random coefficients. Numer. Math. **131**, 329–368 (2015)
23. Gunzburger, M., Webster, C., Zhang, G.: Stochastic finite element methods for partial differential equations with random input data. Acta Numer. **23**, 521–650 (2014)
24. Hackbusch, W.: Hierarchical Matrices: Algorithms and Analysis. Springer, Heidelberg (2015)
25. Haji-Ali, A.L., Nobile, F., Tempone, R.: Multi-index Monte Carlo: when sparsity meets sampling. Numer. Math. **132**, 767–806 (2016)
26. Harbrecht, H., Peters, M., Siebenmorgen, M.: On the quasi-Monte Carlo method with Halton points for elliptic PDEs with log-normal diffusion. Math. Comput. **86**, 771–797 (2017)
27. Herrmann, L., Schwab, Ch.: QMC integration for lognormal-parametric, elliptic PDEs: local supports imply product weights (in review)
28. Joe, S., Kuo, F.Y.: Constructing Sobol' sequences with better two-dimensional projections. SIAM J. Sci. Comput. **30**, 2635–2654 (2008)
29. Kazashi, Y.: Quasi-Monte Carlo integration with product weights for elliptic PDEs with lognormal coefficients. IMA J. Numer. Anal. (to appear)

30. Kuo, F.Y., Nuyens, D.: Hot new directions for quasi-Monte Carlo research in step with applications (in this volume)
31. Kuo, F.Y., Nuyens, D.: Application of quasi-Monte Carlo methods to elliptic PDEs with random diffusion coefficients - a survey of analysis and implementation. Found. Comput. Math. **16**, 1631–1696 (2016)
32. Kuo, F.Y., Schwab, Ch., Sloan, I.H.: Quasi-Monte Carlo finite element methods for a class of elliptic partial differential equations with random coefficient. SIAM J. Numer. Anal. **50**, 3351–3374 (2012)
33. Kuo, F.Y., Schwab, Ch., Sloan, I.H.: Multi-level quasi-Monte Carlo finite element methods for a class of elliptic partial differential equations with random coefficient. Found. Comput. Math. **15**, 411–449 (2015)
34. Kuo, F.Y., Scheichl, R., Schwab, Ch., Sloan, I.H., Ullmann, E.: Multilevel Quasi-Monte Carlo methods for lognormal diffusion problems. Math. Comput. **86**, 2827–2860 (2017)
35. Le Gia, Q.T.: A QMC-spectral method for elliptic PDEs with random coefficients on the unit sphere. In: Dick, J., Kuo, F.Y., Peters, G.W., Sloan, I.H. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2012, pp. 491–508. Springer, Heidelberg (2013)
36. Lemieux, C.: Monte Carlo and Quasi-Monte Carlo Sampling. Springer, New York (2009)
37. Leobacher, G., Pillichshammer, F.: Introduction to Quasi-Monte Carlo Integration and Applications. Springer, Berlin (2014)
38. Niederreiter, H.: Random Number Generation and Quasi-Monte Carlo Methods. SIAM, Philadelphia (1992)
39. Nuyens, D.: The construction of good lattice rules and polynomial lattice rules. In: Kritzer, P., Niederreiter, H., Pillichshammer, F., Winterhof, A. (eds.) Uniform Distribution and Quasi-Monte Carlo Methods. Radon Series on Computational and Applied Mathematics, vol. 15, pp. 223–256. De Gruyter, Berlin (2014)
40. Robbe, P., Nuyens, D., Vandewalle, S.: A multi-index quasi-Monte Carlo algorithm for lognormal diffusion problems. SIAM J. Sci. Comput. Â **39**, S851–S872 (2017)
41. Scheichl, R., Stuart, A., Teckentrup, A.L.: Quasi-Monte Carlo and multilevel Monte Carlo methods for computing posterior expectations in elliptic inverse problems. SIAM/ASA J. Uncertain. Quantif. **5**, 493–518 (2017)
42. Schwab, Ch.: QMC Galerkin discretizations of parametric operator equations. In: Dick, J., Kuo, F.Y., Peters, G.W., Sloan, I.H. (eds.) Monte Carlo and Quasi-Monte Carlo Methods 2012, pp. 613–629. Springer, Heidelberg (2013)
43. Schwab, Ch., Gittelson, C.J.: Sparse tensor discretizations of high-dimensional parametric and stoch astic PDEs. Acta Numer. **20**, 291–467 (2011)
44. Sloan, I.H., Joe, S.: Lattice Methods for Multiple Integration. Oxford University Press, Oxford (1994)
45. Sloan, I.H., Wang, X., Woźniakowski, H.: Finite-order weights imply tractability of multivariate integration. J. Complex. **20**, 46–74 (2004)