Stochastic Simulations

Autumn Semester 2024

Prof. Fabio Nobile *Assistant: Matteo Raviola*

Lab 08 – 7 November 2024

QMC and LH sampling

On randomized QMC formulas

Let $P = \{X_1, \ldots, X_N\}$, $X_i \in \mathbb{R}^d$, be a low-discrepancy sequence and denote the QMC quadrature by $\hat{\mu}_{QMC} = \frac{1}{N}\sum_{i=1}^N \psi(X_i)$. We are interested in estimating the error $|\mu - \hat{\mu}_{QMC}|$. Notice that since the points X_i are *not* i.i.d., we can't use a variance estimator or a CLT as in MC. In order to be able to do this, we can *randomize* the QMC formula. Let $U_j \overset{iid}{\sim} \mathcal{U}([0,1]^d)$, $j = 1, \ldots, K$. If the set of points P is a low discrepancy point set, so is the *randomly shifted point set* $P_{U,j} := \{\{X_1 + U_j\}, \ldots, \{X_N + U_j\}\}$, where $\{\cdot\}$ represents the fractional part. Moreover, since $U_j \overset{iid}{\sim} \mathcal{U}([0,1]^d)$, so is $\{X_i + U_j\}$ for any $i = 1, \ldots, N$. Thus, we can apply a Monte Carlo estimator on $\hat{\mu}_{QMC}$, by computing K independent estimators $\hat{\mu}^j_{QMC}$ for each of the randomly shifted point sets $P_{U,j}$, and then averaging out the estimators. This in turn results in an unbiased estimator $\hat{\hat{\mu}}_{QMC}$ of μ, for which we can use the standard variance estimator and CLT results. C.f the lecture notes for more details.

On generating low-discrepancy sequences

Use the module `sobol_new.py` available on the course's website to generate Sobol sequences.[1] The `Python` [2] syntax `R = generate_points(N,d,0)` generates a matrix `R` of size $N \times d$ corresponding to N vectors of dimension d.

Exercise 1

Consider the problem of approximating the integral

$$I_d(f) = \int_{[0,1]^d} f(\boldsymbol{x})\, d\boldsymbol{x} \ ,$$

for some given function $f \colon [0,1]^d \to \mathbb{R}$. In this exercise we will investigate the approximation qualities of different estimators of $I_d(f)$ for various functions f, which differ mainly by their

[1] These functions were adapted from John Burkardt's website page at the Florida State University: `http://people.sc.fsu.edu/~jburkardt/m_src/m_src.html`. There you can also find many other sequence generators.

[2] Download the files `sobol_new.py` and `Sobol_new-joe-kuo-6.21201` from the course website and use them by writing `from sobol_new import *` at the beginning of your python script. Both files should be in the same directory

regularity. Specifically, for each function listed below address to the following points. Perform all computations at least for $d = 2$ and $d = 20$.

1. Implement a *crude Monte Carlo* estimator to approximate the integral $I_d(f)$.

   Estimate the error using the central limit theorem (CLT). Plot both the exact error (c.f. exact solutions below) and the CLT-based error estimate as functions of the number of used samples M, say, and estimate the convergence rate.

2. Implement a *Latin Hypercube Sampling* estimator using N points in the hypercube to approximate $I_d(f)$.

   Estimate the error using a sample variance estimator based on K repetitions of the Latin Hypercube Sampling estimator. Again, plot both the exact error and an asymptotic confidence interval based error estimate as functions of the number of points N, say, and estimate the convergence rate.

3. Implement a *Quasi Monte Carlo* (QMC) estimator to approximate the integral $I_d(f)$. Use the module `sobol_lib.py` available on the course's website to generate Sobol sequences.

   Estimate the error using the CLT by estimating the variance with a *randomized QMC*. Once again, plot both the exact error and estimated error based on random shifts as functions of the number of N and estimate the convergence rate.

List of functions

Investigate the approximation techniques for $I_d(f)$ mentioned above for the following functions $f\colon [0,1]^d \to \mathbb{R}$, with $\boldsymbol{x} = (x_1, \dots, x_d)$. Please note that a testing suite with several of the function definitions listed below can be found here[3].

1. Oscillatory function: $f(\boldsymbol{x}) = \cos\left(2\pi w_1 + \sum_{j=1}^d c_j x_j\right)$, with $c_j = 9/d$, $w_1 = \frac{1}{2}$.
   The exact solution is:
   $$I_d(f) = \Re\left(e^{i2\pi w_1} \prod_{j=1}^d \frac{1}{ic_j}(e^{ic_j} - 1)\right) ,$$
   where i denotes the imaginary unit and $\Re(z)$ the real part of $z \in \mathbb{C}$.

2. Product peak: $f(\boldsymbol{x}) = \prod_{j=1}^d \left(c_j^{-2} + (x_j - w_j)^2\right)^{-1}$, with $c_j = 7.25/d$ and $w_j = \frac{1}{2}$.
   Exact solution:
   $$I_d(f) = \prod_{j=1}^d c_j \left(\arctan(c_j(1 - w_j)) + \arctan(c_j w_j)\right) .$$

3. Gaussian: $f(\boldsymbol{x}) = \exp\left(-\sum_{j=1}^d c_j^2 (x_j - w_j)^2\right)$, with $c_j = 7.03/d$ and $w_j = \frac{1}{2}$.
   Exact solution:
   $$I_d(f) = \prod_{j=1}^d \frac{\sqrt{\pi}}{2c_j}\Big(\mathrm{erf}\big(c_j(1 - w_j)\big) + \mathrm{erf}(c_j w_j)\Big) .$$

[3] `https://people.math.sc.edu/Burkardt/c_src/testpack/testpack.html`

4. Continuous function: $f(\boldsymbol{x}) = \exp\left(-\sum_{j=1}^{d} c_j |x_j - w_j|\right)$, with $c_j = 2.04/d$ and $w_j = \frac{1}{2}$.
   Exact solution:
$$I_d(f) = \prod_{j=1}^{d} \frac{1}{c_j}\left(2 - e^{-c_j w_j} - e^{-c_j(1-w_j)}\right) .$$
5. Discontinuous function:
$$f(\boldsymbol{x}) = \begin{cases} 0 & \text{if } x_1 > w_1 \text{ or } x_2 > w_2 \\ \exp\left(\sum_{j=1}^{d} c_j x_j\right) & \text{otherwise,} \end{cases}$$
   with $c_j = 4.3/d$, $w_1 = \frac{\pi}{4}$, and $w_2 = \frac{\pi}{5}$.
   Exact solution:
$$I_d(f) = \frac{\prod_{j=3}^{d}(e^{c_j} - 1)}{\prod_{j=1}^{d} c_j}(e^{c_1 w_1} - 1)(e^{c_2 w_2} - 1) .$$
6. Volume of the simplex:
$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \sum_{j=1}^{d} x_j \le 1 \\ 0 & \text{otherwise.} \end{cases}$$
   Exact solution:
$$I_d(f) = \frac{1}{d!} .$$

Exercise 2

Consider the random boundary value problem (BVP)

$$\begin{cases} \big(a(x,\omega)u'(x,\omega)\big)' = 0 \ , & \text{in } (0,L) \ , \\ u(0,\cdot) = 0 \ , & \\ a(L,\cdot)u'(L,\cdot) = 1 \ , & \end{cases}$$

where ω represents an elementary random event, so that $a \equiv a(x,\omega)$ is a random field. The BVP is a simplified model for a linear beam of length L, which is fixed on one side ($x = 0$) and free on the other at which a unit load is applied. Here, the random field a models the beam's spatially varying uncertain material properties. We are interested in quantifying the resulting uncertainty on the beam's displacement at the free end-point. Specifically, we are interested in studying the expected value of the random variable

$$Z \equiv Z(\omega) := u(L,\omega) = \int_0^L \frac{1}{a(x,\omega)}\, dx \ .$$

However, Z is usually not computable for a general elasticity coefficient a. Instead, we consider the computable, approximate random quantity of interest Z_I, which is obtained by approximating the integral by the midpoint rule on a uniform grid,

$$Z_I \equiv Z_I(\omega) := h \sum_{i=1}^{I-1} \frac{1}{a\big(x_i + \frac{h}{2}, \omega\big)} \ ,$$

with $x_i = ih$, $i = 0, \ldots, I \in \mathbb{N}$, and $h = L/I$.

We are interested in approximating $\mathbb{E}[Z_I]$ for $L = 1$ for two different elasticity coefficients:

(i) the random field a is given by

$$a_1(x,\omega) = \mu + \frac{\sigma}{\pi^2}\sum_{n=1}^{d} \frac{\cos(\pi n x)}{n^2} Y_n(\omega) , \quad Y_n(\omega) \sim U(-1,1) \text{ i.i.d.},$$

where $\mu = 1$ and $\sigma = 4$,

(ii) let $a_2(x,\omega) = \exp\big(\kappa(x,\omega)\big)$, where

$$\kappa(x,\omega) = x + \sqrt{2}\sum_{n=1}^{d} \frac{\sin\big((n-\frac{1}{2})\pi x\big)}{(n-\frac{1}{2})\pi} Y_n(\omega) , \quad Y_n(\omega) \sim \mathcal{N}(0,1) \text{ i.i.d.}$$

To that end, approximate $\mathbb{E}[Z_I]$ for various values of d and for various sub-divisions I using (a) a crude Monte Carlo method, (b) Latin Hybercube Sampling (LHS) and (c) Quasi Monte Carlo (QMC) sampling. Use repeated LHS and randomized QMC to estimate the error and provide asymptotic confidence intervals.