Constructing Quasi-Monte Carlo Points With and Without Sensitivity Analysis

Pierre L'Ecuyer and Gregory Seljak

Université de Montréal

SAMO 2025, Grenoble

Monte Carlo (MC) integration

A simulation model produces a random output X. We want to estimate

$$\mu = \mathbb{E}[X] = \mathbb{E}[f(\boldsymbol{U})] = \int_{(0,1)^s} f(\boldsymbol{u})\, d\boldsymbol{u} = \int_0^1 \cdots \int_0^1 f(u_1, \ldots, u_s)\, du_1 \cdots du_s$$

where $f : (0,1)^s \to \mathbb{R}$ and $\boldsymbol{U} = (U_1, \ldots, U_s)$ is a uniform r.v. over $(0,1)^s$.

(Crude) Monte Carlo method:

1. Generate n independent copies of $\boldsymbol{U} \sim \mathcal{U}(0,1)^s$, say $\boldsymbol{U}_1, \ldots, \boldsymbol{U}_n$;
2. Estimate μ by $\hat{\mu}_n = \frac{1}{n} \sum_{i=1}^n f(\boldsymbol{U}_i)$.

Monte Carlo (MC) integration

A simulation model produces a random output X. We want to estimate

$$\mu = \mathbb{E}[X] = \mathbb{E}[f(\boldsymbol{U})] = \int_{(0,1)^s} f(\boldsymbol{u})\, d\boldsymbol{u} = \int_0^1 \cdots \int_0^1 f(u_1, \ldots, u_s)\, du_1 \cdots du_s$$

where $f : (0,1)^s \to \mathbb{R}$ and $\boldsymbol{U} = (U_1, \ldots, U_s)$ is a uniform r.v. over $(0,1)^s$.

(Crude) Monte Carlo method:

1. Generate n independent copies of $\boldsymbol{U} \sim \mathcal{U}(0,1)^s$, say $\boldsymbol{U}_1, \ldots, \boldsymbol{U}_n$;
2. Estimate μ by $\hat{\mu}_n = \frac{1}{n}\sum_{i=1}^n f(\boldsymbol{U}_i)$.

Almost sure convergence as $n \to \infty$ (strong law of large numbers).
Can use central limit theorem to compute a confidence interval on μ:

$$\mathbb{P}\left[\mu \in \left(\hat{\mu}_n - \frac{c_\alpha S_n}{\sqrt{n}},\ \hat{\mu}_n + \frac{c_\alpha S_n}{\sqrt{n}}\right)\right] \approx 1 - \alpha,$$

where S_n^2 is any consistent estimator of $\sigma^2 = \mathrm{Var}[f(\boldsymbol{U})]$. Converges at slow rate: $\mathcal{O}(n^{-1/2})$.

Quasi-Monte Carlo (QMC)

Replace the independent random points $\boldsymbol{U}_i$ by a set of **deterministic** points $P_n = \{\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{n-1}\}$ that cover $(0,1)^s$ more evenly. Approximate

$$\mu = \mu(f) = \int_{(0,1)^s} f(\boldsymbol{u}) \mathrm{d}\boldsymbol{u} \qquad \text{by} \qquad \bar{\mu}_n = \frac{1}{n} \sum_{i=0}^{n-1} f(\boldsymbol{u}_i).$$

Quasi-Monte Carlo (QMC)

Replace the independent random points $\boldsymbol{U}_i$ by a set of **deterministic** points $P_n = \{\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{n-1}\}$ that cover $(0,1)^s$ more evenly. Approximate

$$\mu = \mu(f) = \int_{(0,1)^s} f(\boldsymbol{u}) \mathrm{d}\boldsymbol{u} \qquad \text{by} \qquad \bar{\mu}_n = \frac{1}{n} \sum_{i=0}^{n-1} f(\boldsymbol{u}_i).$$

For various spaces $\mathcal{H}$ of functions f, one has

$$|E_n| := |\bar{\mu}_n - \mu| \leq \mathcal{D}_n(P_n) \cdot \mathcal{V}(f) \qquad \text{(worst-case error bound)}$$

for all $f \in \mathcal{H}$, where $\mathcal{V}(f)$ is the variation of f in $\mathcal{H}$ and $\mathcal{D}(P_n)$ measures the discrepancy between the empirical distribution of P_n and the uniform distribution over $(0,1)^s$.

Both depend on $\mathcal{H}$. Typically, we know how to construct P_n such that $\mathcal{D}_n(P_n) = \mathcal{O}(n^{-\alpha}(\log n)^{s-1})$ for some $\alpha > 1/2$ (low discrepancy points).
When $\mathcal{V}(f) < \infty$, $|E_n|$ also converges at this rate, which beats the MC rate.

Difficulty: very hard to estimate $\mathcal{V}(f)$ and the error E_n in practice.

Randomized quasi-Monte Carlo (RQMC)

We randomize P_n into $\tilde{P}_n = \{\boldsymbol{U}_1, \ldots, \boldsymbol{U}_n\} \subset (0,1)^s$ such that
(1) the uniformity of P_n as a whole is preserved;
(2) each point $\boldsymbol{U}_i$ has the uniform distribution over $(0,1)^s$.

Unbiased RQMC estimator of μ: $\hat{\mu}_{n,\mathrm{rqmc}} = \dfrac{1}{n}\sum_{i=0}^{n-1} f(\boldsymbol{U}_i)$.

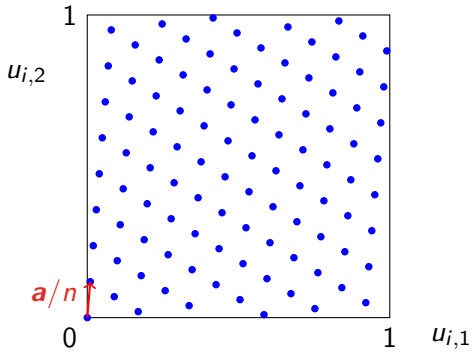
Variance estimation: Make r independent replicates of the RQMC estimator $\hat{\mu}_{n,\mathrm{rqmc}}$, then estimate μ and $\mathrm{Var}[\hat{\mu}_{n,\mathrm{rqmc}}]$ by their sample mean and sample variance.
Variance of sample average is often $\mathcal{O}(r^{-1}n^{-2\alpha+\epsilon})$.

Confidence interval on μ: CIs based on the Student t distribution are usually reliable.

Lattice rule of rank 1

Select integer n and $\boldsymbol{a} = (a_1, \ldots, a_s) \in \{0, 1, \ldots, n-1\}^s$, with $\text{pgcd}(n, a_j) = 1$
Let $\boldsymbol{u}_i = (i/n)\boldsymbol{a} \bmod 1$ for $i = 0, \ldots, n-1$.

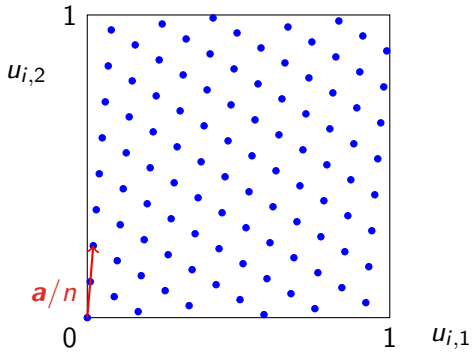
Example with $s = 2$, $n = 101$, $\boldsymbol{a} = (1, 12)$:



Lattice rule of rank 1

Select integer n and $\boldsymbol{a} = (a_1, \ldots, a_s) \in \{0, 1, \ldots, n-1\}^s$, with $\text{pgcd}(n, a_j) = 1$
Let $\boldsymbol{u}_i = (i/n)\boldsymbol{a} \bmod 1$ for $i = 0, \ldots, n-1$.

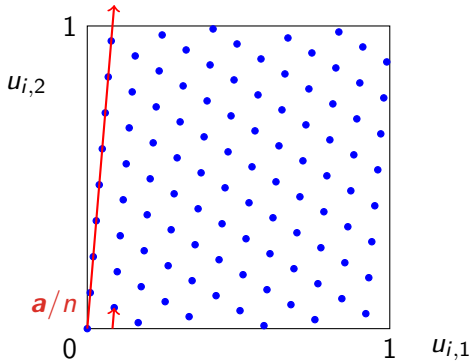
Example with $s = 2$, $n = 101$, $\boldsymbol{a} = (1, 12)$:



Lattice rule of rank 1

Select integer n and $\boldsymbol{a} = (a_1, \ldots, a_s) \in \{0, 1, \ldots, n-1\}^s$, with $\text{pgcd}(n, a_j) = 1$
Let $\boldsymbol{u}_i = (i/n)\boldsymbol{a} \bmod 1$ for $i = 0, \ldots, n-1$.

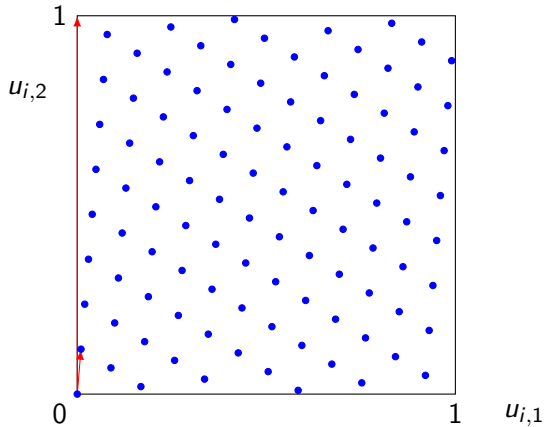
Example with $s = 2$, $n = 101$, $\boldsymbol{a} = (1, 12)$:



Randomly-Shifted Lattice

Random shift modulo 1: Replace P_n by $(P_n + \boldsymbol{U}) \bmod 1$ where $\boldsymbol{U} \sim U[0,1)^s$.
This preserves the lattice structure and satisfies the RQMC conditions.

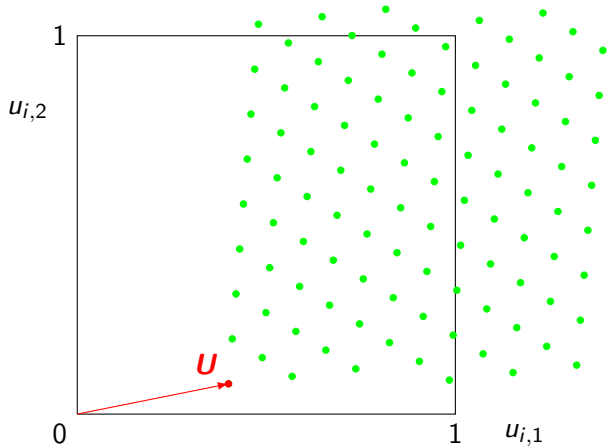
Example: lattice with $s = 2$, $n = 101$, $\boldsymbol{a} = (1, 12)$



Randomly-Shifted Lattice

Random shift modulo 1: Replace P_n by $(P_n + \boldsymbol{U}) \bmod 1$ where $\boldsymbol{U} \sim U[0,1)^s$.
This preserves the lattice structure and satisfies the RQMC conditions.

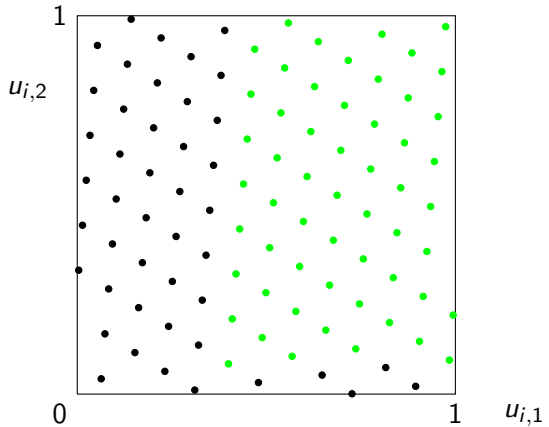
Example: lattice with $s = 2$, $n = 101$, $\boldsymbol{a} = (1, 12)$



Randomly-Shifted Lattice

Random shift modulo 1: Replace P_n by $(P_n + \boldsymbol{U}) \bmod 1$ where $\boldsymbol{U} \sim U[0,1)^s$.
This preserves the lattice structure and satisfies the RQMC conditions.

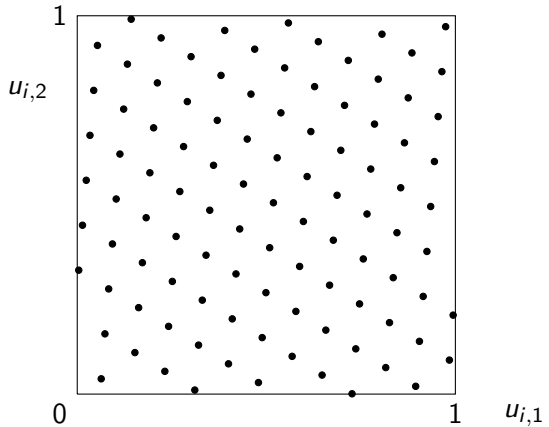
Example: lattice with $s = 2$, $n = 101$, $\boldsymbol{a} = (1, 12)$



Randomly-Shifted Lattice

Random shift modulo 1: Replace P_n by $(P_n + \boldsymbol{U}) \bmod 1$ where $\boldsymbol{U} \sim U[0,1)^s$.
This preserves the lattice structure and satisfies the RQMC conditions.

Example: lattice with $s = 2$, $n = 101$, $\boldsymbol{a} = (1, 12)$



Variance in a space of one-periodic smooth functions

We consider one-periodic smooth functions f with Fourier expansion

$$f(\boldsymbol{u}) = \sum_{\boldsymbol{h} \in \mathbb{Z}^s} \hat{f}(\boldsymbol{h}) \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{h}^{\mathrm{t}} \boldsymbol{u}} \qquad \text{for } \boldsymbol{u} \in [0,1)^s.$$

For a randomly shifted lattice, $\mathrm{Var}[\hat{\mu}_{n,\mathrm{rqmc}}] = \sum_{\boldsymbol{0} \neq \boldsymbol{h} \in L_s^*} |\hat{f}(\boldsymbol{h})|^2$, with L_s^* the dual lattice.

Variance in a space of one-periodic smooth functions

We consider one-periodic smooth functions f with Fourier expansion

$$f(\boldsymbol{u}) = \sum_{\boldsymbol{h}\in\mathbb{Z}^s} \hat{f}(\boldsymbol{h}) e^{2\pi \mathrm{i} \boldsymbol{h}^{\mathrm{t}}\boldsymbol{u}} \qquad \text{for } \boldsymbol{u} \in [0,1)^s.$$

For a randomly shifted lattice, $\mathrm{Var}[\hat{\mu}_{n,\mathrm{rqmc}}] = \sum_{\boldsymbol{0}\neq\boldsymbol{h}\in L_s^*} |\hat{f}(\boldsymbol{h})|^2$, with L_s^* the dual lattice.

For an integer $\alpha > 0$, we consider a class of functions f with **smoothness** α, i.e., with square-integrable mixed partial derivatives up to order α, and for which the periodic continuation of its derivatives up to order $\alpha - 1$ is **continuous across the unit cube boundaries**. For such functions, we know how to construct a sequence of rank-1 lattices (a vector $\boldsymbol{a} = \boldsymbol{a}(n)$ for each n) for which QMC Error $= \mathcal{O}(\mathcal{P}_\alpha)$ and

$$\mathrm{Var}[\hat{\mu}_{n,\mathrm{rqmc}}] = \mathcal{O}(\mathcal{P}_{2\alpha}) \quad \text{where} \quad \mathcal{P}_\alpha := \sum_{\boldsymbol{0}\neq\boldsymbol{h}\in L_s^*} \prod_{j:h_j\neq 0} h_j^{-\alpha} = \mathcal{O}(n^{-\alpha+\epsilon}) \text{ for any } \epsilon > 0.$$

Weighted $\mathcal{P}_{\gamma,\alpha}$ with projection-dependent weights $\gamma_{\mathfrak{u}}$

Denote $\mathfrak{u}(\boldsymbol{h}) = \mathfrak{u}(h_1, \ldots, h_s)$ the set of indices j for which $h_j \neq 0$.

$$\mathcal{P}_{\gamma,\alpha} \quad = \sum_{\mathbf{0} \neq \boldsymbol{h} \in L_s^*} \gamma_{\mathfrak{u}(\boldsymbol{h})} \prod_{j \in \mathfrak{u}(\boldsymbol{h})} h_j^{-\alpha}.$$

For α integer > 0, with $\boldsymbol{u}_i = (u_{i,1}, \ldots, u_{i,s}) = \boldsymbol{a}/n \bmod 1$, we have a finite sum:

$$\mathcal{P}_{\gamma,2\alpha} = \sum_{\emptyset \neq \mathfrak{u} \subseteq \{1,\ldots,s\}} \gamma_{\mathfrak{u}} \left[\frac{-(-4\pi^2)^{\alpha}}{(2\alpha)!}\right]^{|\mathfrak{u}|} \frac{1}{n} \sum_{i=0}^{n-1} \prod_{j \in \mathfrak{u}} B_{2\alpha}(u_{i,j}),$$

and the corresponding variation is

$$\mathcal{V}_{\gamma,\alpha}^2(f) = \sum_{\emptyset \neq \mathfrak{u} \subseteq \{1,\ldots,s\}} \frac{1}{\gamma_{\mathfrak{u}}(4\pi^2)^{\alpha|\mathfrak{u}|}} \int_{[0,1]^{|\mathfrak{u}|}} \left| \frac{\partial^{\alpha|\mathfrak{u}|}}{\partial \boldsymbol{u}^{\alpha}} f_{\mathfrak{u}}(\boldsymbol{u}) \right|^2 d\boldsymbol{u},$$

for $f : [0,1)^s \to \mathbb{R}$ smooth enough. Then, we have the variance bound

$$\mathrm{Var}[\hat{\mu}_{n,\mathrm{rqmc}}] = \sum_{\mathfrak{u} \subseteq \{1,\ldots,s\}} \mathrm{Var}[\hat{\mu}_{n,\mathrm{rqmc}}(f_{\mathfrak{u}})] \;\leq\; \mathcal{V}_{\gamma,\alpha}^2(f) \cdot \mathcal{P}_{\gamma,2\alpha}\,.$$

We also know how to construct lattices for which $\mathcal{P}_{\gamma,2\alpha} = \mathcal{O}(n^{-2\alpha+\epsilon})$ for any $\epsilon > 0$.

We also know how to construct lattices for which $\mathcal{P}_{\gamma,2\alpha} = \mathcal{O}(n^{-2\alpha+\epsilon})$ for any $\epsilon > 0$.

But how do we find appropriate weights $\gamma_{\mathfrak{u}}$?
This $\mathcal{P}_{\gamma,2\alpha}$ is the RQMC variance for the worst-case function with $\mathcal{V}^2_{\gamma,\alpha}(f) \leq 1$:

$$f^*(\boldsymbol{u}) = \sum_{\mathfrak{u}\subseteq\{1,\ldots,s\}} \sqrt{\gamma_{\mathfrak{u}}} \prod_{j\in\mathfrak{u}} \frac{(2\pi)^\alpha}{(\alpha)!} B_\alpha(u_j).$$

For this worst-case function, we have

$$\sigma^2_{\mathfrak{u}} = \gamma_{\mathfrak{u}} \left[\mathrm{Var}[B_\alpha(U)] \frac{(4\pi^2)^\alpha}{((\alpha)!)^2}\right]^{|\mathfrak{u}|} = \gamma_{\mathfrak{u}} \left[|B_{2\alpha}(0)| \frac{(4\pi^2)^\alpha}{(2\alpha)!}\right]^{|\mathfrak{u}|}.$$

For $\alpha = 1$, we should take $\gamma_{\mathfrak{u}} = (3/\pi^2)^{|\mathfrak{u}|}\sigma^2_{\mathfrak{u}} \approx (0.30396)^{|\mathfrak{u}|}\sigma^2_{\mathfrak{u}}$.
For $\alpha = 2$, we should take $\gamma_{\mathfrak{u}} = [45/\pi^4]^{|\mathfrak{u}|}\sigma^2_{\mathfrak{u}} \approx (0.46197)^{|\mathfrak{u}|}\sigma^2_{\mathfrak{u}}$.
For $\alpha \to \infty$, we have $\gamma_{\mathfrak{u}} \to (0.5)^{|\mathfrak{u}|}\sigma^2_{\mathfrak{u}}$.

If we have estimates of the variance components $\sigma^2_{\mathfrak{u}}$, we can compute and use the corresponding weights $\gamma_{\mathfrak{u}}$. But this is often costly to compute in practice!

Random generating vectors

What if we just pick $\boldsymbol{a}$ at random instead of optimizing it?

Main advantage: No need to estimate variance components and select the weights.

Random generating vectors

What if we just pick $\boldsymbol{a}$ at random instead of optimizing it?

Main advantage: No need to estimate variance components and select the weights.

Algorithm:

Repeat r times:

Draw random $\boldsymbol{a}$ uniformly in $\{\boldsymbol{a} : 1 < a_j < n \text{ and } \gcd(a_j, n) = 1 \text{ for all } j\}$.

Compute QMC or RQMC estimator with corresponding lattice rule.

Return average or median of these r estimators.

Random generating vectors

What if we just pick $\boldsymbol{a}$ at random instead of optimizing it?

Main advantage: No need to estimate variance components and select the weights.

Algorithm:

Repeat r times:

Draw random $\boldsymbol{a}$ uniformly in $\{\boldsymbol{a} : 1 < a_j < n \text{ and } \gcd(a_j, n) = 1 \text{ for all } j\}$.

Compute QMC or RQMC estimator with corresponding lattice rule.

Return average or median of these r estimators.

It turns out that most choices of $\boldsymbol{a}$ are pretty good.
Very few are very bad and yield a huge variance.
By taking the median instead of the mean, we can essentially remove their impact.

Proven results: With the median, the QMC error is $\mathcal{O}(n^{-\alpha+\epsilon})$ with probability $> 1 - p$ where p decreases exponentially in r.

Goda and L (2022): For a deterministic lattice rule with random generating vector, for any $\epsilon > 0$, $0 < \rho < 1$, smoothness α, and choice of weights γ (to define the function space), there is a constant c_1 for which

$$\mathbb{P}\left[\sup_{f \in \mathcal{F},\, \|f\| \le 1} |\mu(f) - Q_{\mathrm{med}}(f)| > \frac{c_1}{(n\rho)^{\alpha-\epsilon}}\right] \le \frac{\rho^{(r+1)/2}}{4}$$

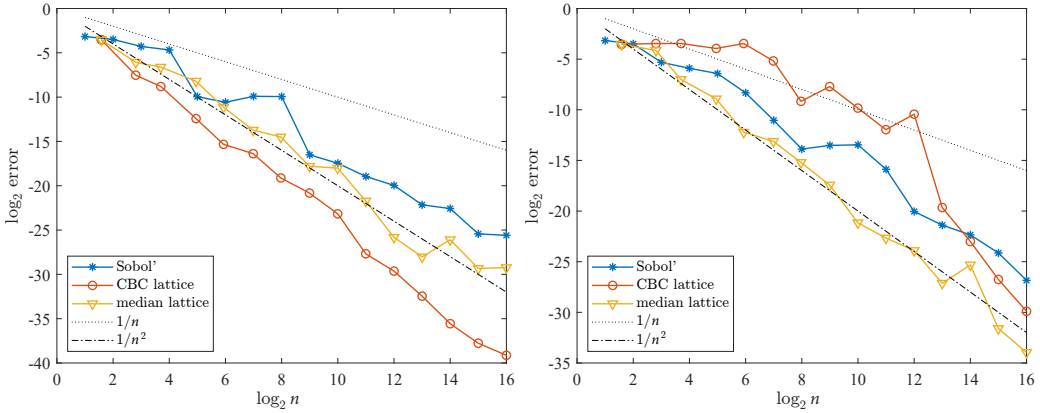
where $Q_{\mathrm{med}}(f)$ is the median estimator.

Similar result for polynomial lattice rules.

Example: Error for a periodic integrand

$$f(\boldsymbol{u}) = \prod_{j=1}^{s} \left[1 + j^{-3}\left(30u_j^2(1-u_j)^2 - 1\right)\right]$$

On the left, we compare the median rule with $r = 11$ vs CBC with proper weights.
On the right, the coordinates of $\boldsymbol{u}$ were reversed, so CBC had wrong weights.



Digital net in base 2

Gives $n = 2^k$ QMC points in s dimensions.

Select two integers $w \geq k \geq 0$, and s generating matrices $\boldsymbol{C}_1, \cdots, \boldsymbol{C}_s$ of dimensions $w \times k$, with binary entries, and whose first k rows are linearly independent. For $i = 0, \ldots, b^k - 1$ and $j = 1, \ldots, s$, let:

$$
\begin{aligned}
i &= a_{i,0} + a_{i,1}2 + \cdots + a_{i,k-1}2^{k-1} && (k \text{ input digits in base } 2) \\
\begin{pmatrix} u_{i,j,1} \\ \vdots \\ u_{i,j,w} \end{pmatrix} &= \boldsymbol{C}_j \begin{pmatrix} a_{i,0} \\ \vdots \\ a_{i,k-1} \end{pmatrix} \bmod 2, && (w \text{ output digits in base } 2) \\
u_{i,j} &= \sum_{\ell=1}^{w} u_{i,j,\ell} 2^{-\ell}, && \boldsymbol{u}_i = (u_{i,1}, \ldots, u_{i,s}) \in [0,1)^s.
\end{aligned}
$$

Random digital shift

Generate $\boldsymbol{U} \sim U(0,1)^s$ and XOR it bitwise with each $\boldsymbol{u}_i$. Sufficient for RQMC.

Additional randomization

Most common: left matrix srambling (LMS): For Sobol' upper-triangular matrices $\boldsymbol{C}_j$ ($b = 2$), generate random lower-triangular matrices $\boldsymbol{L}_j$, usually with $w > k$, to obtain the new generating matrices $\tilde{\boldsymbol{C}}_j = \boldsymbol{L}_j \boldsymbol{C}_j$. These $\tilde{\boldsymbol{C}}_j$ inherit the equidistribution properties of the $\boldsymbol{C}_j$.

$$\boldsymbol{L}_j = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ \ell_{2,1} & 1 & 0 & \dots & 0 \\ & \vdots & & \ddots & 0 \\ \ell_{k,1} & \ell_{k,2} & & & 1 \\ \ell_{k+1,1} & \ell_{k+1,2} & & & \ell_{k+1,k} \\ & \vdots & & & \vdots \\ \ell_{w,1} & \ell_{w,2} & & & \ell_{w,k} \end{pmatrix} \quad \text{and} \quad \boldsymbol{C}_j = \begin{pmatrix} 1 & v_{1,2} & \dots & v_{1,k} \\ 0 & 1 & \dots & v_{2,k} \\ \vdots & 0 & \ddots & \vdots \\ & \vdots & & 1 \end{pmatrix}.$$

This only changes the $\boldsymbol{C}_j$'s. After that, we must apply a random digital shift to get RQMC. Adding LMS provably reduces the variance to $O(n^{-3}(\log n)^s)$ when f is sufficiently smooth, compared with $O(n^{-2}(\log n)^s)$ for Sobol' points with a random digital shift alone.

In a way, LMS is similar to randomizing the generating vector $\boldsymbol{a}$ in lattice rules. We could also try to "optimize" the $\boldsymbol{L}_j$'s instead of taking them at random; see L'Ecuyer et al. (2024).

Median estimator for LMS + digital shift

Pan and Owen (2023, 2024) show that for the average of r RQMC replicates with LMS for Sobol' points, the MSE cannot converge faster than $\mathcal{O}(n^{-3}r^{-1})$.
This is because some rare realizations of L_j are very bad (e.g., row $k+1$ is all zero).

Median estimator for LMS + digital shift

Pan and Owen (2023, 2024) show that for the average of r RQMC replicates with LMS for Sobol' points, the MSE cannot converge faster than $\mathcal{O}(n^{-3}r^{-1})$.
This is because some rare realizations of L_j are very bad (e.g., row $k+1$ is all zero).

By taking the median, the probability that such realizations have an impact is negligible.

For the median of r RQMC replicates, for analytic f, if $w \geq \lambda k^2/s$ and $r = \Omega(k^2)$, we get super-polynomial convergence:

$$\mathrm{MSE}(\mathrm{median}) = \mathcal{O}(n^{-2c\log_2(n)/s}) \text{ for } c \approx 0.21.$$

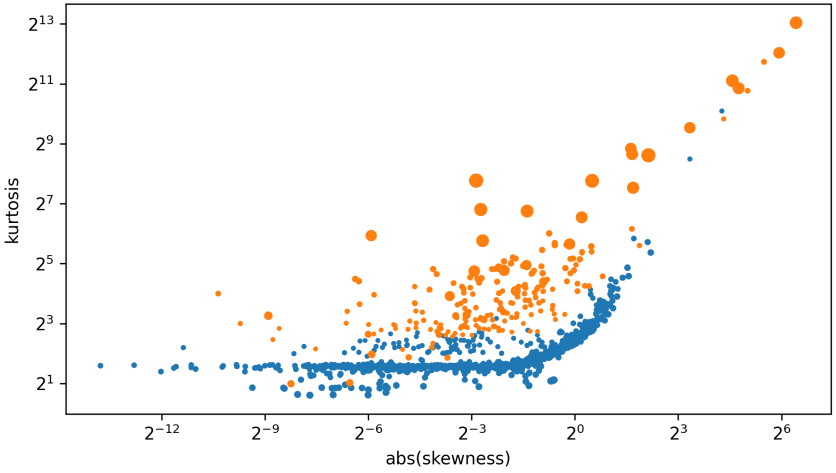
The median is better in this case because the RQMC estimator has a very large kurtosis.

When is the median better than the mean?

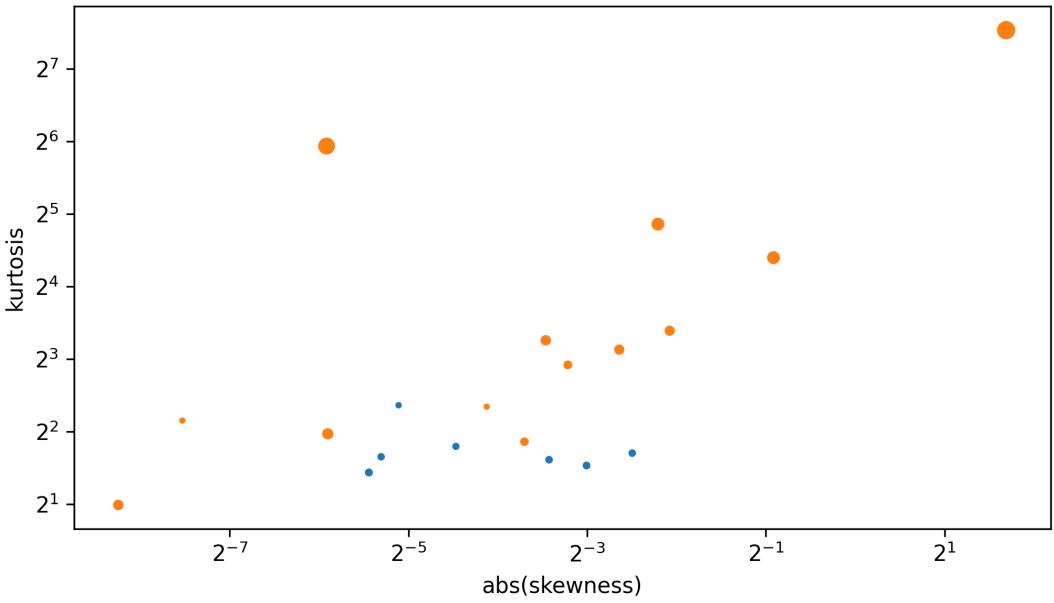
We tried 600 test cases (different f, s, n, RQMC method), with $r = 11$ and 31.

orange when ratio MSE(mean)/ MSE(median) is > 1, **blue** otherwise.

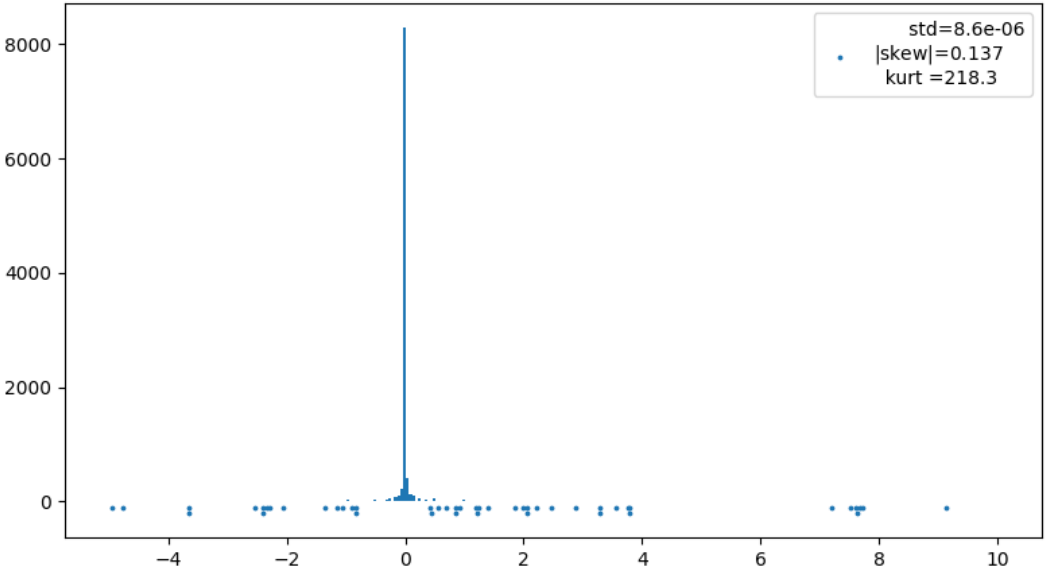
Size of point is proportional to $|\log(\text{ratio})|$. The largest ratio is near 8,000.

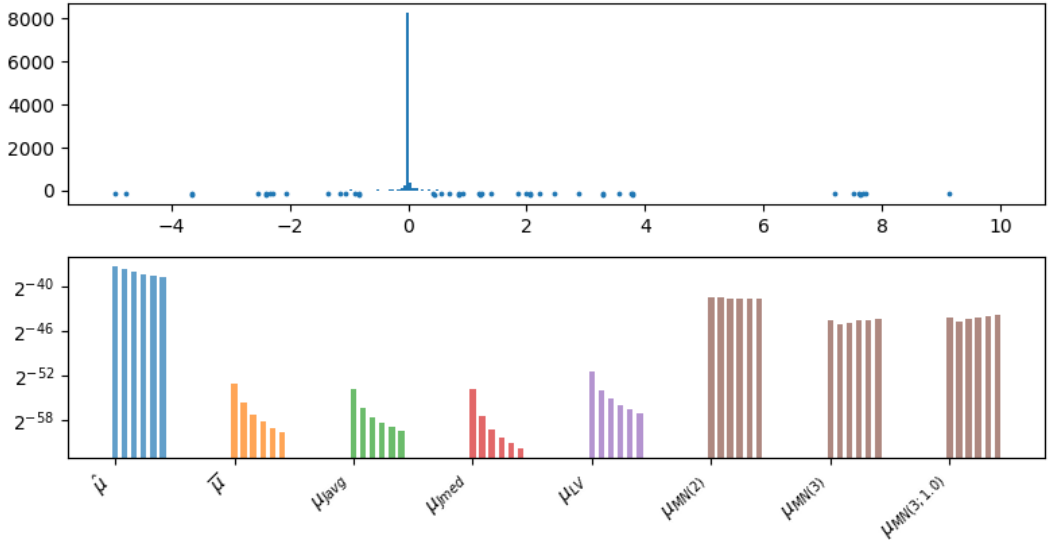


One specific case: MC2 function, Sobol + LMS



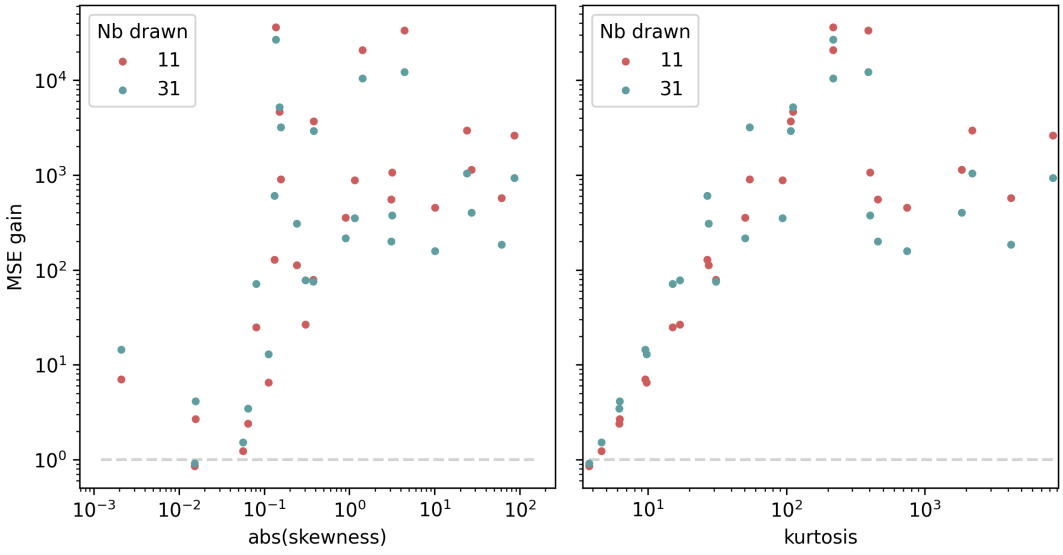
SumUeU function, Sobol + LMS, $s = 8$, $n = 2^{12}$, 10^4 obs. of RQMC estimator



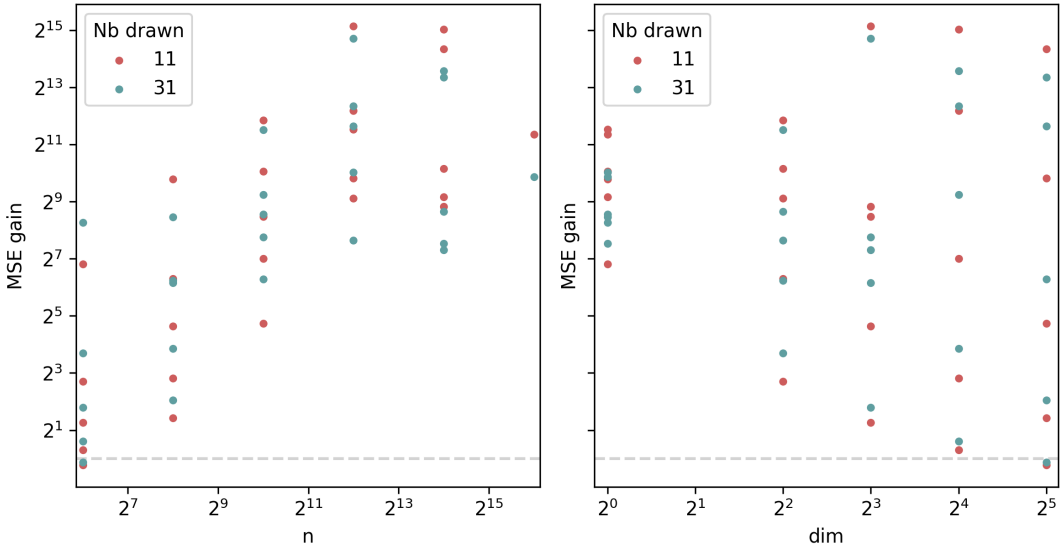


MSE for each estimator (r=[11, 15, 19, 23, 27, 31])

SumUeU function, Sobol + LMS



SumUeU function, Sobol + LMS



Conclusions

- We can avoid searching for good QMC parameters based on weights that depend on the sensitivities of projections, just by sampling the parameters randomly.
- This may lead to an RQMC estimator with high kurtosis, in which case it is safer to take the median instead of the mean.
- Very simple to implement, no need to estimate proper weights, but more difficult to estimate the error and compute a confidence interval.
- On the other hand, when we can compute good QMC parameters, it is usually better to take the mean, not the median.

Thank you!

References

- J. Dick and F. Pillichshammer. *Digital Nets and Sequences: Discrepancy Theory and Quasi-Monte Carlo Integration*. Cambridge Univ. Press, 2010.
- T. Goda and P. L'Ecuyer, Construction-free median quasi-Monte Carlo rules for function spaces with unspecified smoothness and general weights, SIAM Journal on Scientific Computing, 44, 4 (2022), A2765-A2788.
- T. Goda, K. Suzuki, and M. Matsumoto. A universal median quasi-Monte Carlo integration. SIAM Journal on Num. Analysis, 62 (1): 533–566, 2024.
- P. L'Ecuyer, Y. Cherkanihassani, and M. E. A. Derkaoui. Pre-scrambled digital nets for randomized quasi-Monte Carlo. In *Proceedings of the 2024 Winter Simulation Conference*, 443–454, 2024.
- P. L'Ecuyer, M. Nakayama, A. B. Owen, and B. Tuffin. Confidence Intervals for Randomized Quasi-Monte Carlo Estimators, *Proceedings of the 2023 Winter Simulation Conference*, IEEE Press, pages 445–456, 2023.
- P. L'Ecuyer. Quasi-Monte Carlo methods with applications in finance. Finance and Stochastics, 13(3):307–349, 2009.
- P. L'Ecuyer. Randomized quasi-Monte Carlo: An introduction for practitioners. In P. W. Glynn and A. B. Owen, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2016*, 2017.
- P. L'Ecuyer, P. Marion, M. Godin, and F. Puchhammer. A Tool for Custom Construction of QMC and RQMC Point Sets. *Monte Carlo and Quasi-Monte Carlo Methods 2020*, A. Keller, Ed., Springer-Verlag, 51–70, 2022.
- P. L'Ecuyer and D. Munger. Algorithm 958: Lattice builder: A general software tool for constructing rank-1 lattice rules. ACM Trans. on Mathematical Software, 42(2):Article 15, 2016.
- P. L'Ecuyer and D. Munger. Constructing adapted lattice rules using problem-dependent criteria. In *Proceedings of the 2012 Winter Simulation Conference*, pages 373–384. IEEE Press, 2012.
- A. B. Owen. *Practical Quasi-Monte Carlo*. Draft available at `https://artowen.su.domains/mc/practicalqmc.pdf`, 2023.
- Z. Pan and A. B. Owen. Super-polynomial accuracy of one dimensional randomized nets using the median of means. Mathematics of Computation, 92(340):805–837, 2023.
- Z. Pan and A. B. Owen. Super-polynomial accuracy of multidimensional randomized nets using the median of means. Mathematics of Computation, 93(349):2265–2289, 2024.