The Journal of Open Source Software

QMCPy: A Python Framework for (Quasi-)Monte Carlo Algorithms

Aleksei G. Sorokin [1,2], **Fred J. Hickernell** [1], **Sou-Cheng T. Choi** [1,3], **Jagadeeswaran Rathinavel** [4], **Pieterjan Robbe** [5], and **Aadit Jain** [6]

1 Illinois Institute of Technology, USA **2** University of Chicago, USA **3** SouLab LLC, USA **4** Torc Robotics, USA **5** Sandia National Laboratories, USA **6** University of California San Diego, USA

DOI: 10.21105/joss.09705

Software

- Review
- Repository
- Archive

Editor: Owen Lockwood

Reviewers:

- @matt-graham
- @tupui

Submitted: 20 November 2025
Published: 19 January 2026

Summary

Monte Carlo (MC) methods estimate high-dimensional integrals by computing sample averages at independent and identically distributed (IID) random points. Quasi-Monte Carlo (QMC) methods replace IID samples with low-discrepancy (LD) sequences which more uniformly cover the integration domain, leading to faster convergence and reduced computational requirements. Figure 1 visualizes IID and LD sequences.

QMCPy (https://qmcsoftware.github.io/QMCSoftware) (Choi et al., 2026) is our Python package for high-dimensional numerical integration using MC and QMC methods, collectively "(Q)MC." Its object-oriented design enables researchers to easily implement novel (Q)MC algorithms. The framework offers user-friendly APIs, diverse (Q)MC algorithms, adaptive error estimation techniques, and integration with scientific libraries following reproducible research practices (Choi et al., 2022; Choi, 2014). Compared to previous versions, QMCPy v2.2 (which is easily installed with `pip install -U qmcpy`) includes

- improved documentation,
- strengthened tests and demos, and
- expanded support for randomized LD sequences.

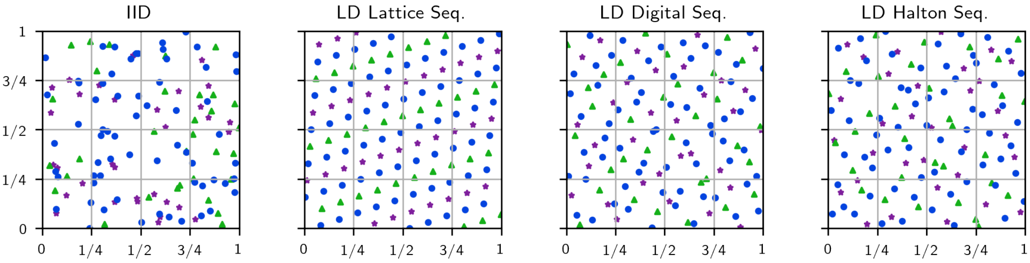

Figure 1: An IID sequence with gaps and clusters alongside LD sequences with more uniform coverage. Each sequence contains purple stars (initial 32 points), green triangles (next 32), and blue circles (subsequent 64).

Statement of Need

(Q)MC methods are essential for computational finance (Giles & Waterhouse, 2009; Lemieux, 2004; X. Wang & Sloan, 2005; Zhang et al., 2021), uncertainty quantification (Kaarnioja et al., 2021; Marzouk et al., 2016; Parno et al., 2014, 2021; Seelinger et al., 2023), machine learning (Chen et al., 2018; Dick & Feischl, 2021), and physics (Albert & Barabási, 2002; Bernhard et al., 2015; Landau & Binder, 2014). While (Q)MC methods are well established (Dick et al., 2013; Dick & Pillichshammer, 2010), practical implementation demands numerical and algorithmic expertise. QMCPy follows MATLAB's Guaranteed Automatic Integration Library (GAIL) (Choi et al., 2021; Tong et al., 2022) in consolidating a broad range of cutting-edge (Q)MC algorithms into a unified framework (Choi et al., 2022, 2024; Hickernell et al., 2026; Sorokin, 2025; Sorokin & Rathinavel, 2022). QMCPy features

- **intuitive APIs** for (Q)MC components,
- **flexible integrations** with NumPy (Harris et al., 2020), SciPy (Virtanen et al., 2020), and PyTorch (Paszke et al., 2019),
- **robust and adaptive sampling** with theoretically grounded error estimation, and
- **extensible components** enabling researchers to implement and test new algorithms.

While popular modules like scipy.stats.qmc (Roy et al., 2023) and torch.quasirandom (Paszke et al., 2019) provide basic (Q)MC sequences such as Sobol' and Halton, QMCPy provides (Q)MC researchers and practitioners an end-to-end framework with additional capabilities to enable state-of-the-art (Q)MC techniques. Advanced features unique to QMCPy include

- customizable LD sequences with diverse randomization techniques,
- efficient generators of LD sequences with multiple independent randomizations,
- automatic variable transformations for (Q)MC compatibility, and
- rigorous adaptive error estimation algorithms.

Components

(Q)MC methods approximate the multivariate integral

$$\mu := \mathbb{E}[g(\mathbf{T})] = \int_{\mathcal{T}} g(\mathbf{t})\, \lambda(\mathbf{t})\, d\mathbf{t}, \qquad \mathbf{T} \sim \lambda, \tag{1}$$

where g is the **integrand** and λ is the probability density of a random variable $\mathbf{T}$ whose distribution we call the **true measure**. To accommodate LD samples (approximately uniform on $[0,1]^d$), a transformation ψ is performed to rewrite μ as

$$\mu = \mathbb{E}[f(\mathbf{X})] = \int_{[0,1]^d} f(\mathbf{x})\, d\mathbf{x}, \qquad \mathbf{X} \sim \mathcal{U}[0,1]^d. \tag{2}$$

If $\mathbf{T} \sim \psi(\mathbf{X})$, then $f = g \circ \psi$.

(Q)MC methods estimate the population mean μ in (2) via the sample mean

$$\hat{\mu} := \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{X}_i). \tag{3}$$

MC methods use IID $\mathbf{X}_1, \dots, \mathbf{X}_n$ and have error $|\hat{\mu} - \mu|$ like $\mathcal{O}(n^{-1/2})$ (Niederreiter, 1978). QMC methods choose dependent LD nodes that fill $[0,1]^d$ more evenly, i.e., the discrepancy between the **discrete distribution** of $\mathbf{X}_1, \dots, \mathbf{X}_n$ and the uniform distribution is small. QMC methods can achieve errors like $\mathcal{O}(n^{-1+\delta})$ where $\delta > 0$ is arbitrarily small (Hickernell & Wang, 2002; X. Wang, 2003). A key feature of QMCPy is **stopping criteria** that automatically determine n so $|\mu - \hat{\mu}| \le \varepsilon$ for a user-specified tolerance $\varepsilon > 0$, deterministically or with high probability.

QMCPy contains four main abstract classes:

1. **Discrete Distributions** generate IID or randomized LD sequences (Sorokin, 2025) including
   - **Lattices** with random shifts (Coveyou & MacPherson, 1967; Cranley & Patterson, 1976; Hickernell et al., 2005; Richtmyer, 1951; Y. Wang & Hickernell, 2002).
   - **Digital Sequences** (including Sobol' and Faure constructions) with digital shifts (DS), linear matrix scrambling (LMS), or nested uniform scrambling (NUS, also

    - called Owen scrambling) (Dick, 2011; Dick & Pillichshammer, 2005; Dick & Pillichshammer, 2010; Matoušek, 1998; Niederreiter, 1987, 1992; Owen, 1995, 2003; Sobol', 1967). Higher-order digital sequences are available to enable QMC convergence like $\mathcal{O}(n^{-\alpha+\delta})$ when f has α degrees of smoothness (Dick, 2011).
    - **Halton Sequences** with digital permutations, DS, LMS, or NUS (Halton, 1960; Matoušek, 1998; Morokoff & Caflisch, 1994; Owen & Pan, 2024; X. Wang & Hickernell, 2000).

   Internally, QMCPy's LD generators call our C package QMCToolsCL (Sorokin, 2026). We also integrate with the LDData repository (Sorokin et al., 2025) which collects lattice generating vectors and digital sequence generating matrices from Kuo's websites (Cools et al., 2006; Joe & Kuo, 2003, 2010; Joe & Kuo, 2008; Kuo, 2007; Nuyens & Cools, 2006), the Magic Point Shop (Kuo & Nuyens, 2016), and LatNet Builder (L'Ecuyer et al., 2022).

2. **True Measures** come with default transformations ψ satisfying $\psi(\mathbf{X}) \sim \mathbf{T}$. For example, if $\mathbf{T} \sim \mathcal{N}(\mathbf{m}, \Sigma = \mathbf{A}\mathbf{A}^T)$ is a d-dimensional Gaussian, then $\psi(\mathbf{X}) = \mathbf{A}\Phi^{-1}(\mathbf{X}) + \mathbf{m}$ where Φ^{-1} is the inverse Gaussian distribution function applied elementwise. We support the broad range of measures included in scipy.stats (Virtanen et al., 2020).

3. **Integrands** g, given a transformation ψ, automatically set $f = g \circ \psi$ so that $\mu = \mathbb{E}[g(\mathbf{T})] = \mathbb{E}[f(\mathbf{X})]$.

4. **Stopping Criteria (SC)** adaptively increase the sample size n until (Q)MC estimates satisfy user-defined error tolerances (Hickernell, Choi, et al., 2018; Owen, 2024; Tong et al., 2022). SC include guaranteed MC algorithms (Hickernell et al., 2013) and QMC algorithms based on:

    - multiple randomizations of LD sequences (L'Ecuyer et al., 2023),
    - quickly tracking the decay of Fourier coefficients (Ding et al., 2020; Hickernell, Jiménez Rugama, et al., 2018; Hickernell & Jiménez Rugama, 2016; Jiménez Rugama & Hickernell, 2016), or
    - fast Bayesian cubature (Rathinavel, 2019; Rathinavel & Hickernell, 2019, 2022).

   QMCPy is also capable of simultaneously approximating functions of multiple integrands (Sorokin & Rathinavel, 2022), and we are actively expanding support for multilevel (Q)MC algorithms following Julia's MultilevelEstimators.jl (Robbe, 2024).

   Figure 2 compares (Q)MC SC for Asian option pricing with 100 independent trials per error tolerance ε. The left and middle plots show median lines and shaded regions for 10%–90% quantiles. While MC SC require $n = \mathcal{O}(1/\varepsilon^2)$ samples (and time), QMC SC require only $n = \mathcal{O}(1/\varepsilon)$. (Q)MC SC consistently meet tolerances, with the right plot showing distributions of errors for a single error tolerance.

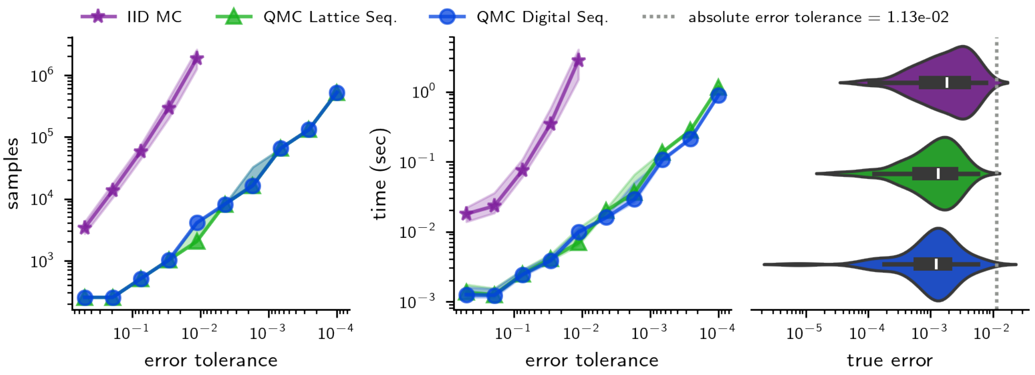

Figure 2: (Q)MC SC for Asian option pricing.

Acknowledgements

The authors acknowledge support from the U.S. National Science Foundation grant DMS-2316011 and the Department of Energy Office of Science Graduate Student Research Program. We thank the international (Q)MC research community as well as JOSS reviewers and editors for invaluable and timely feedback and support.

This article has been co-authored by employees of National Technology and Engineering Solutions of Sandia, LLC under Contract No. DE-NA0003525 with the U.S. Department of Energy (DOE). The employees co-own right, title and interest in and to the article and are responsible for its contents. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this article or allow others to do so, for United States Government purposes. The DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (https://www.energy.gov/downloads/doe-public-access-plan).

AS is partially supported by DARPA The Right Space HR0011-25-9-0031.

References

Albert, R., & Barabási, A.-L. (2002). Statistical mechanics of complex networks. *Reviews of Modern Physics*, *74*(1), 47–97. https://doi.org/10.1103/RevModPhys.74.47

Bernhard, J. E., Marcy, P. W., Coleman-Smith, C. E., Huzurbazar, S., Wolpert, R. L., & Bass, S. A. (2015). Quantifying properties of hot and dense QCD matter through systematic model-to-data comparison. *Physical Review C*, *91*(5), 054910. https://doi.org/10.1103/physrevc.91.054910

Chen, W. Y., Mackey, L., Gorham, J., Briol, F.-X., & Oates, C. (2018). Stein points. In J. Dy & A. Krause (Eds.), *Proceedings of the 35th International Conference on Machine Learning* (Vol. 80, pp. 844–853). PMLR. https://proceedings.mlr.press/v80/chen18f.html

Choi, S.-C. T. (2014). MINRES-QLP Pack and reliable reproducible research via staunch scientific software. *Journal of Open Research Software*, *2*(1), 1–7. https://doi.org/10.5334/jors.bb

Choi, S.-C. T., Ding, Y., Hickernell, F. J., Jiang, L., Jiménez Rugama, Ll. A., Li, D., Rathinavel, J., Tong, X., Zhang, K., Zhang, Y., & Zhou, X. (2021). *GAIL: Guaranteed Automatic Integration Library (versions 1.0–2.3.2)*. https://doi.org/10.5281/zenodo.4018189

Choi, S.-C. T., Ding, Y., Hickernell, F. J., Rathinavel, J., & Sorokin, A. G. (2024). Challenges in developing great quasi-Monte Carlo software. *Monte Carlo and Quasi-Monte Carlo Methods: MCQMC, Linz, Austria, July 2022*, 209–222. https://doi.org/10.1007/978-3-031-59762-6_9

Choi, S.-C. T., Hickernell, F. J., Rathinavel, J., McCourt, M. J., & Sorokin, A. G. (2022). Quasi-Monte Carlo software. In A. Keller (Ed.), *Monte Carlo and Quasi-Monte Carlo Methods: MCQMC, Oxford, England, August 2020* (pp. 23–50). Springer, Cham. https://doi.org/10.1007/978-3-030-98319-2_2

Choi, S.-C. T., Hickernell, F. J., Rathinavel, J., McCourt, M., & Sorokin, A. G. (2026). *QMCPy: A quasi-Monte Carlo Python library*. Python Software, Zenodo. https://doi.org/10.5281/zenodo.3964489

Cools, R., Kuo, F. Y., & Nuyens, D. (2006). Constructing embedded lattice rules for multivariate integration. *SIAM Journal on Scientific Computing*, *28*(6), 2162–2188. https://doi.org/10.1137/06065074x

Coveyou, R., & MacPherson, R. D. (1967). Fourier analysis of uniform random number generators. *Journal of the ACM (JACM)*, *14*(1), 100–119. https://doi.org/10.1145/321371.321379

Cranley, R., & Patterson, T. N. L. (1976). Randomization of number theoretic methods for multiple integration. *SIAM Journal on Numerical Analysis*, *13*, 904–914. https://doi.org/10.1137/0713071

Dick, J. (2011). Higher order scrambled digital nets achieve the optimal rate of the root mean square error for smooth integrands. *The Annals of Statistics*, *39*(3), 1372–1398. https://doi.org/10.1214/11-AOS880

Dick, J., & Feischl, M. (2021). A quasi-Monte Carlo data compression algorithm for machine learning. *Journal of Complexity*, *67*, 101587. https://doi.org/10.1016/j.jco.2021.101587

Dick, J., Kuo, F. Y., & Sloan, I. H. (2013). High-dimensional integration: The quasi-Monte Carlo way. *Acta Numerica*, *22*, 133–288. https://doi.org/10.1017/s0962492913000044

Dick, J., & Pillichshammer, F. (2005). Multivariate integration in weighted Hilbert spaces based on Walsh functions and weighted Sobolev spaces. *Journal of Complexity*, *21*(2), 149–195. https://doi.org/10.1016/j.jco.2004.07.003

Dick, J., & Pillichshammer, F. (2010). *Digital nets and sequences: Discrepancy theory and quasi-Monte Carlo integration*. Cambridge University Press. https://doi.org/10.1017/CBO9780511761188

Ding, Y., Hickernell, F. J., & Jiménez Rugama, Ll. A. (2020). An adaptive algorithm employing continuous linear functionals. In P. L'Ecuyer & B. Tuffin (Eds.), *Monte Carlo and Quasi-Monte Carlo Methods: MCQMC, Rennes, France, July 2018* (Vol. 324, pp. 161–181). Springer, Cham. https://doi.org/10.1007/978-3-030-43465-6_8

Giles, M. B., & Waterhouse, B. J. (2009). Multilevel quasi-Monte Carlo path simulation. *Advanced Financial Modelling, Radon Series on Computational and Applied Mathematics*, *8*, 165–181. https://doi.org/10.1515/9783110213140.165

Halton, J. H. (1960). On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, *2*, 84–90. https://doi.org/10.1007/bf01386213

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hickernell, F. J., Choi, S.-C. T., Jiang, L., & Jiménez Rugama, Ll. A. (2018). Monte Carlo simulation, automatic stopping criteria for. In M. Davidian, B. Everitt, R. S. Kenett, G. Molenberghs, W. Piegorsch, & F. Ruggeri (Eds.), *Wiley StatsRef-Statistics Reference Online*. John Wiley & Sons Ltd. https://doi.org/10.1002/9781118445112.stat08035

Hickernell, F. J., Jiang, L., Liu, Y., & Owen, A. B. (2013). Guaranteed conservative fixed width confidence intervals via Monte Carlo sampling. In J. Dick, F. Y. Kuo, G. W. Peters, & I. H. Sloan (Eds.), *Monte Carlo and quasi-Monte Carlo methods 2012* (Vol. 65, pp. 105–128). Springer-Verlag, Berlin. https://doi.org/10.1007/978-3-642-41095-6_5

Hickernell, F. J., & Jiménez Rugama, Ll. A. (2016). Reliable adaptive cubature using digital sequences. In R. Cools & D. Nuyens (Eds.), *Monte Carlo and quasi-Monte Carlo methods: MCQMC, Leuven, Belgium, April 2014* (Vol. 163, pp. 367–383). Springer-Verlag, Berlin. https://doi.org/10.1007/978-3-319-33507-0_18

Hickernell, F. J., Jiménez Rugama, Ll. A., & Li, D. (2018). Adaptive quasi-Monte Carlo methods for cubature. In J. Dick, F. Y. Kuo, & H. Woźniakowski (Eds.), *Contemporary*

Computational Mathematics — a celebration of the 80th birthday of Ian Sloan (pp. 597–619). Springer-Verlag. https://doi.org/10.1007/978-3-319-72456-0_27

Hickernell, F. J., Kirk, N., & Sorokin, A. G. (2026). Quasi-Monte Carlo methods: What, why, and how? In C. Lemieux & B. Feng (Eds.), *Monte Carlo and Quasi-Monte Carlo Methods: MCQMC, Waterloo, Canada, August 2024*. Accepted for publication; Springer, Cham. https://doi.org/10.48550/arXiv.2502.03644

Hickernell, F. J., Lemieux, C., & Owen, A. B. (2005). Control variates for quasi-Monte Carlo. *Statistical Science*, *20*, 1–31. https://doi.org/10.1214/088342304000000468

Hickernell, F. J., & Wang, X. (2002). The error bounds and tractability of quasi-Monte Carlo algorithms in infinite dimension. *Mathematics of Computation*, *71*, 1641–1661. https://doi.org/10.1090/S0025-5718-01-01377-1

Jiménez Rugama, Ll. A., & Hickernell, F. J. (2016). Adaptive multidimensional integration based on rank-1 lattices. In R. Cools & D. Nuyens (Eds.), *Monte Carlo and quasi-Monte Carlo methods: MCQMC, Leuven, Belgium, April 2014* (Vol. 163, pp. 407–422). Springer-Verlag, Berlin. https://doi.org/10.1007/978-3-319-33507-0_20

Joe, S., & Kuo, F. Y. (2003). Remark on algorithm 659: Implementing Sobol's quasirandom sequence generator. *ACM Transactions on Mathematical Software*, *29*, 49–57. https://doi.org/10.1145/641876.641879

Joe, S., & Kuo, F. Y. (2008). Constructing Sobol sequences with better two-dimensional projections. *SIAM Journal on Scientific Computing*, *30*(5), 2635–2654. https://doi.org/10.1137/070709359

Joe, S., & Kuo, F. Y. (2010). *Sobol' sequence generator*. https://web.maths.unsw.edu.au/~fkuo/sobol/

Kaarnioja, V., Kazashi, Y., Kuo, F. Y., Nobile, F., & Sloan, I. H. (2021). Fast approximation by periodic kernel-based lattice-point interpolation with application in uncertainty quantification. *Numerische Mathematik*, *150*, 33–77. https://doi.org/10.1007/s00211-021-01242-3

Kuo, F. Y. (2007). *Lattice rule generating vectors*. https://web.maths.unsw.edu.au/~fkuo/lattice/

Kuo, F. Y., & Nuyens, D. (2016). Application of quasi-Monte Carlo methods to elliptic PDEs with random diffusion coefficients – a survey of analysis and implementation. *Foundations of Computational Mathematics*, *16*, 1631–1696. https://doi.org/10.1007/s10208-016-9329-5

L'Ecuyer, P., Marion, P., Godin, M., & Puchhammer, F. (2022). A tool for custom construction of QMC and RQMC point sets. In A. Keller (Ed.), *Monte Carlo and Quasi-Monte Carlo Methods: MCQMC, Oxford, England, August 2020* (pp. 51–70). Springer, Cham. https://doi.org/10.1007/978-3-030-98319-2_3

L'Ecuyer, P., Nakayama, M. K., Owen, A. B., & Tuffin, B. (2023). Confidence intervals for randomized quasi-Monte Carlo estimators. *2023 Winter Simulation Conference (WSC)*, 445–456. https://doi.org/10.1109/wsc60868.2023.10408613

Landau, D. P., & Binder, K. (2014). *A guide to Monte Carlo simulations in statistical physics*. Cambridge University Press. https://doi.org/10.1017/9781108780346

Lemieux, C. (2004). Randomized quasi-Monte Carlo: A tool for improving the efficiency of simulations in finance. In R. G. Ingalls, M. D. Rossetti, J. S. Smith, & B. A. Peters (Eds.), *Proceedings 2004 Winter Simulation Conference* (pp. 1565–1573). IEEE Press. https://doi.org/10.1109/wsc.2004.1371499

Marzouk, Y., Moselhy, T., Parno, M., & Spantini, A. (2016). Sampling via measure transport: An introduction. In R. Ghanem, D. Higdon, & H. Owhadi (Eds.), *Handbook of Uncertainty*

Quantification (pp. 1–41). Springer International Publishing. https://doi.org/10.1007/978-3-319-11259-6_23-1

Matoušek, J. (1998). On the L_2-discrepancy for anchored boxes. *Journal of Complexity*, *14*, 527–556. https://doi.org/10.1006/jcom.1998.0489

Morokoff, W. J., & Caflisch, R. E. (1994). Quasi-random sequences and their discrepancies. *SIAM Journal on Scientific Computing*, *15*, 1251–1279. https://doi.org/10.1137/0915077

Niederreiter, H. (1978). Quasi-Monte Carlo methods and pseudo-random numbers. *Bull. Amer. Math. Soc.*, *84*, 957–1041. https://doi.org/10.1090/s0002-9904-1978-14532-7

Niederreiter, H. (1987). Point sets and sequences with small discrepancy. *Monatshefte Für Mathematik*, *104*, 273–337. https://doi.org/10.1007/bf01294651

Niederreiter, H. (1992). *Random number generation and quasi-Monte Carlo methods*. SIAM.

Nuyens, D., & Cools, R. (2006). Fast algorithms for component-by-component construction of rank-1 lattice rules in shift-invariant reproducing kernel Hilbert spaces. *Mathematics of Computation*, *75*(254), 903–920. https://doi.org/10.1090/s0025-5718-06-01785-6

Owen, A. B. (1995). Randomly permuted (t, m, s)-nets and (t, s)-sequences. In H. Niederreiter & P. J.-S. Shiue (Eds.), *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing* (Vol. 106, pp. 299–317). Springer-Verlag, New York. https://doi.org/10.1007/978-1-4612-2552-2_19

Owen, A. B. (2003). Variance with alternative scramblings of digital nets. *ACM Transactions of Modeling and Computer Simulation*, *13*(4). https://doi.org/10.1145/945511.945518

Owen, A. B. (2024). *Error estimation for quasi-Monte Carlo*. https://doi.org/10.48550/arXiv.2501.00150

Owen, A. B., & Pan, Z. (2024). Gain coefficients for scrambled Halton points. *SIAM Journal on Numerical Analysis*, *62*(3), 1021–1038. https://doi.org/10.1137/23m1601882

Parno, M., Davis, A., & Seelinger, L. (2021). MUQ: The MIT Uncertainty Quantification library. *Journal of Open Source Software*, *6*(68), 3076. https://doi.org/10.21105/joss.03076

Parno, M., Davis, A., Seelinger, L., & Marzouk, Y. (2014). *MIT uncertainty quantification (MUQ) library*. https://mituq.bitbucket.io/source/_site/index.html

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., … Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32* (pp. 8024–8035). Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Rathinavel, J. (2019). *Fast automatic Bayesian cubature using matching kernels and designs* [PhD thesis, Illinois Institute of Technology]. http://hdl.handle.net/10560/islandora:1009768

Rathinavel, J., & Hickernell, F. J. (2019). Fast automatic Bayesian cubature using lattice sampling. *Statistics and Computing*, *29*, 1215–1229. https://doi.org/10.1007/s11222-019-09895-9

Rathinavel, J., & Hickernell, F. J. (2022). Fast automatic Bayesian cubature using Sobol' sampling. In Z. Botev, A. Keller, C. Lemieux, & B. Tuffin (Eds.), *Advances in Modeling and Simulation: Festschrift in Honour of Pierre L'Ecuyer* (pp. 301–318). Springer, Cham. https://doi.org/10.1007/978-3-031-10193-9_15

Richtmyer, R. D. (1951). *The evaluation of definite integrals, and a Quasi-Monte-Carlo method based on the properties of algebraic numbers* (LA-1342). Los Alamos Scientific Laboratory.

https://doi.org/10.2172/4405295

Robbe, P. (2024). *MultilevelEstimators.jl: A Julia package for multilevel Monte Carlo*. https://github.com/PieterjanRobbe/MultilevelEstimators.jl

Roy, P. T., Owen, A. B., Balandat, M., & Haberland, M. (2023). Quasi-Monte Carlo methods in Python. *Journal of Open Source Software*, *8*(84), 5309. https://doi.org/10.21105/joss.05309

Seelinger, L., Cheng-Seelinger, V., Davis, A., Parno, M., & Reinarz, A. (2023). UM-Bridge: Uncertainty quantification and modeling bridge. *Journal of Open Source Software*, *8*(83), 4748. https://doi.org/10.21105/joss.04748

Sobol', I. M. (1967). 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. https://doi.org/10.1016/0041-5553(67)90144-9

Sorokin, A. G. (2025). *QMCPy: A Python software for randomized low-discrepancy sequences, quasi-Monte Carlo, and fast kernel methods*. https://doi.org/10.48550/arXiv.2502.14256

Sorokin, A. G. (2026). *QMCToolsCL: Quasi-Monte Carlo sequence generators in OpenCL and C*. https://github.com/QMCSoftware/QMCToolsCL

Sorokin, A. G., Hickernell, F. J., & L'Ecuyer, P. (2025). *LDData: Low discrepancy generating vectors and matrices*. https://github.com/QMCSoftware/LDData

Sorokin, A. G., & Rathinavel, J. (2022). On bounding and approximating functions of multiple expectations using quasi-Monte Carlo. *International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, 583–599. https://doi.org/10.1007/978-3-031-59762-6_29

Tong, X., Choi, S.-C. T., Ding, Y., Hickernell, F. J., Jiang, L., Jiménez Rugama, Ll. A., Rathinavel, J., Zhang, K., Zhang, Y., & Zhou, X. (2022). Guaranteed automatic integration library (GAIL): An open-source MATLAB library for function approximation, minimization, and integration. *Journal of Open Research Software*, *10*, 7. https://doi.org/10.5334/jors.381

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2

Wang, X. (2003). Strong tractability of multivariate integration using Quasi-Monte Carlo algorithms. *Mathematics of Computation*, *72*, 823–838. https://doi.org/10.1090/s0025-5718-02-01440-0

Wang, X., & Hickernell, F. J. (2000). Randomized Halton sequences. *Mathematical and Computer Modelling*, *32*, 887–899. https://doi.org/10.1016/s0895-7177(00)00178-3

Wang, X., & Sloan, I. H. (2005). Why are high-dimensional finance problems often of low effective dimension? *SIAM J. Sci. Comput.*, *27 (1)*, 159–183. https://doi.org/10.1137/s1064827503429429

Wang, Y., & Hickernell, F. J. (2002). An historical overview of lattice point sets. In K. T. Fang, F. J. Hickernell, & H. Niederreiter (Eds.), *Monte Carlo and quasi-Monte Carlo methods 2000* (pp. 158–167). Springer-Verlag, Berlin. https://doi.org/10.1007/978-3-642-56046-0_10

Zhang, Y., Li, J., Wang, H., & Choi, S.-C. T. (2021). Sentiment-guided adversarial learning for stock price prediction. *Frontiers in Applied Mathematics and Statistics*, *7*, 8. https://doi.org/10.3389/fams.2021.601105