Challenges in Developing Great MCQMC Software

Fred J. Hickernell
Department of Applied Mathematics & Center for Interdisciplinary Scientific Computation
Illinois Institute of Technology `hickernell@iit.edu` `mypages.iit.edu/~hickernell`

with input from Mark Klinchin, the GAIL an QMCPy teams, and friends
partially supported by SigOpt, an Intel company

Please join us Tuesday, July 19, for lunch at ???
to discuss the future of MCQMC software

Thanks to the organizers as we meet again in person
Slides at `speakerdeck.com/fjhickernell/quasi-monte-carlo-software`

MCQMC 2022, Linz, July 18, 2022

My Definition of Great MCQMC Software

- Correct, e.g., not omit the zeroth point in a low discrepancy sequence (Owen 2022; SciPy Developers 2020)
- Complete—contain the components or easily access components in other libraries to solve real, complex problems
- Accessible—tutorials, demos, discussion forums, etc. for (new) users; written in a language that potential users speak; provide a consistent user interface
- Efficient—in terms of compute time and memory
- Current—include the latest and best algorithms
- Sustainable—have a sufficient user base and developer community for updates and maintenance
- Scalable—take advantage of advanced computer architectures for speed and to solve large problems

Selective History of MC**QMC** Software

- L'Ecuyer (2017) provides a history of random number generation
- Notorious randu from the 1960s (Wikipedia n.d.), which fails the spectral test
- VEGAS (Lepage 1978; Lepage 2021) importance sampling Monte Carlo algorithm favored by physicists
- ACM low discrepancy point generators (Bratley and Fox 1988; Bratley, Fox, and Niederreiter 1992; Hong and H. 2003)
- FinDer (Paskov and Traub 1995; Papageorgiou n.d.) and BRODA (Kucherenko n.d.) targeting quantitative finance
- Korobov cubature and scrambled Sobol′ generators in NAG (The Numerical Algorithms Group 2021) for decades
- Scrambled Sobol′ and Halton generators in MATLAB (The MathWorks, Inc. 2022) since 2008, and fixed a few years later

Selective History of MC**QMC** Software

- Stan (Stan Development Team 2022) for Markov Chain Monte Carlo (MCMC)
- Keller
- L'Ecuyer
- Nuyens
- Hickernell
- TensorFlow
- Scrambled Sobol' in SciPy (Virtanen, Gommers, Oliphant, Haberland, Reddy, Cournapeau, Burovski, Peterson, Weckesser, et al. 2020) and PyTorch (Paszke, Gross, Massa, Lerer, Bradbury, Chanan, Killeen, Lin, Gimelshein, et al. 2019)

Why Do We Need Software?

Software Library Architecture

content...

Choice of Language

content...

Advanced Architectures

content...

Introductory Demos

content...

Connecting Demos

content...

Building User & Developer Ownership

content...

From Private to Public

content...

Next Steps

Prof. Dr. Gerhard Kette
Aubrunnerweg
5K Film & Drohnenaufnahmen...
Aubrunnerweg
Aubrunnerweg
Mengerstraße
Bootshaus Loisach
Die* Obelisk
Universität
Pizzeria Bella Casa
Aubrunnerweg
China-Restaurant Jade Garten
Schnell geliefert
Mengerstraße
Altenberger Straße
Hartlauer
Optician
Europagymnasium Auhof
Ella Ladestation

References I

- Bratley, P. and B. L. Fox (1988). “Algorithm 659: Implementing Sobol’s Quasirandom Sequence Generator”. In: *ACM Trans. Math. Software* 14, pp. 88–100.
- Bratley, P., B. L. Fox, and H. Niederreiter (1992). “Implementation and Tests of Low-Discrepancy Sequences”. In: *ACM Trans. Model. Comput. Simul.* 2, pp. 195–213.
- Hong, H. S. and F. J. H. (2003). “Algorithm 823: Implementing Scrambled Digital Nets”. In: *ACM Trans. Math. Software* 29, pp. 95–109. DOI: `10.1145/779359.779360`.
- Kucherenko, S. (n.d.). *BRODA*. URL: `https://www.broda.co.uk/index.html`.
- L’Ecuyer, P. (2017). “History of uniform random number generation”. In: *Proceedings of the 2017 Winter Simulation Conference*. Ed. by W. K. V. Chan et al. DOI: `10.1109/WSC.2017.8247790`.
- Lepage, G. P. (1978). “A new algorithm for adaptive multidimensional integration”. In: *J. Comput. Phys.* 27, pp. 192–203.
- — (2021). “Adaptive multidimensional integration: VEGAS enhanced”. In: *J. Comput. Phys.* 439, p. 110386.

References II

- Owen, A. B. (2022). "On dropping the first Sobol′ point". In: *Monte Carlo and Quasi-Monte Carlo Methods: MCQMC, Oxford, England, August 2020*, pp. 71–86. URL: https://arxiv.org/pdf/2008.08051.pdf.
- Papageorgiou, A. (n.d.). *FinDer*. URL: http://www.cs.columbia.edu/~ap/html/finder.html.
- Paskov, S. and J. Traub (1995). "Faster Valuation of Financial Derivatives". In: *J. Portfolio Management* 22, pp. 113–120.
- Paszke, Adam et al. (2019). "PyTorch: An imperative style, high-performance deep learning library". In: *Advances in neural information processing systems* 32, pp. 8026–8037.
- SciPy Developers (2020). *scipy discussion of Sobol′ sequence implementation*. URL: https://github.com/scipy/scipy/pull/10844.
- Stan Development Team (2022). *Stan Modeling Language Users Guide and Reference Manual, version 2.30*. URL: http://mc-stan.org.
- The MathWorks, Inc. (2022). *MATLAB R2022b*. Natick, MA.

References III

- The Numerical Algorithms Group (2021). *The NAG Library*. Mark 27. Oxford.
- Virtanen, Pauli et al. (2020). “SciPy 1.0: fundamental algorithms for scientific computing in Python”. In: *Nature Methods* 17.3, pp. 261–272.
- Wikipedia (n.d.). *RANDU*. URL: `https://en.wikipedia.org/wiki/RANDU`.