

Analysis of pseudo-random number generators in QMC-SSE method

Dong-Xu Liu(刘东旭), Wei Xu(徐维), and Xue-Feng Zhang(张学锋)
Citation: Chin. Phys. B, 2024, 33 (**3**): 037509. DOI: 10.1088/1674-1056/ad1e69
Journal homepage: http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Magnetic excitations of diagonally coupled checkerboards

Tingting Yan(颜婷婷), Shangjian Jin(金尚健), Zijian Xiong(熊梓健), Jun Li(李军), and Dao-Xin Yao(姚道新)
Chin. Phys. B, 2021, 30 (**10**): 107505. DOI: 10.1088/1674-1056/ac1b94

Emergent O(4) symmetry at the phase transition from plaquette-singlet to antiferromagnetic order in quasi-two-dimensional quantum magnets

Guangyu Sun(孙光宇), Nvsen Ma(马女森), Bowen Zhao(赵博文), Anders W. Sandvik, and Zi Yang Meng(孟子杨)
Chin. Phys. B, 2021, 30 (**6**): 067505. DOI: 10.1088/1674-1056/abf3b8

Tunable deconfined quantum criticality and interplay of different valence-bond solid phases

Bowen Zhao(赵博文), Jun Takahashi, Anders W. Sandvik
Chin. Phys. B, 2020, 29 (**5**): 057506. DOI: 10.1088/1674-1056/ab889b

Role of the spin anisotropy of the interchain interaction in weakly coupled antiferromagnetic Heisenberg chains

Yuchen Fan(樊宇辰), Rong Yu(俞榕)
Chin. Phys. B, 2020, 29 (**5**): 057505. DOI: 10.1088/1674-1056/ab820b

Typicality at quantum-critical points

Lu Liu(刘录), Anders W Sandvik, Wenan Guo(郭文安)
Chin. Phys. B, 2018, 27 (**8**): 087501. DOI: 10.1088/1674-1056/27/8/087501

COMPUTATIONAL PROGRAMS FOR PHYSICS

Analysis of pseudo-random number generators in QMC-SSE method

Dong-Xu Liu(刘东旭), Wei Xu(徐维), and Xue-Feng Zhang(张学锋)†

Department of Physics, Chongqing University, Chongqing 401331, *China*

(Received 13 November 2023; revised manuscript received 9 January 2024; accepted manuscript online 15 January 2024)

In the quantum Monte Carlo (QMC) method, the pseudo-random number generator (PRNG) plays a crucial role in determining the computation time. However, the hidden structure of the PRNG may lead to serious issues such as the breakdown of the Markov process. Here, we systematically analyze the performance of different PRNGs on the widely used QMC method known as the stochastic series expansion (SSE) algorithm. To quantitatively compare them, we introduce a quantity called QMC efficiency that can effectively reflect the efficiency of the algorithms. After testing several representative observables of the Heisenberg model in one and two dimensions, we recommend the linear congruential generator as the best choice of PRNG. Our work not only helps improve the performance of the SSE method but also sheds light on the other Markov-chain-based numerical algorithms.

Keywords: stochastic series expansion, quantum Monte Carlo, pseudo-random number generator

PACS: 75.40.Mg, 02.70.Ss, 75.30.Kz **DOI:** 10.1088/1674-1056/ad1e69

1. Introduction

The Monte Carlo method is one of the most popular algorithms in various science and technology fields,[1] and its key use is to produce many independent samples following a specific distribution. To simulate the quantum many-body system, various quantum Monte Carlo (QMC) algorithms have been invented and improved.[2] One of the most popular methods is the stochastic series expansion (SSE) method.[3–6] The target distribution is the partition function represented with a series expanded form

$$Z = \mathrm{Tr}\left\{ \mathrm{e}^{-\beta H} \right\} = \sum_{\alpha} \sum_{n=0}^{\infty} \frac{(-\beta)^n}{n!} \langle \alpha | H^n | \alpha \rangle , \tag{1}$$

where H is the Hamiltonian simulated and $\beta = 1/T$ is the inverse of the temperature. Then, the updating processes of the SSE algorithm specifically depend on the form of the Hamiltonian. However, all the random sampling processes require the pseudo-random number generator (PRNG). Therefore, it is clear that the performance of the QMC highly relies on the suitability of the PRNG.[7–9]

The PRNG can generate a lot of numbers that behave as though randomly distributed,[10] but the "pseudo" part of its name means the numbers do not originate from truly random physical processes, such as the quantum effect, thermal fluctuation, or chaos. However, because the sequence of the random numbers $\{s_0, s_1, s_2, \ldots, s_i, \ldots\}$ presents very low autocorrelation, it can still be used to simulate random processes. Usually, PRNGs working on a computer can be separated into the following steps: (i) defining the iteration function, which can calculate s_{i+1} based on the constant parameters and previous number s_i; (ii) initializing the parameters of the iteration function and also the first number s_0 (sometimes equal to "seed", which is the specific initial condition of the iteration function); and (iii) calculating the random number with the help of the iteration function step by step. The PRNG is pseudo-random, so it has some intrinsic problems, such as the periodicity $s_{p+n} = s_n$ with period p, or the "lattice" structure that sometimes exists.[11]

The effect of PRNGs on the Monte Carlo algorithm has been studied in depth. For the classical Monte Carlo, different PRNGs were tested for various updating algorithms like Metropolis, Wolff, and Swendsen–Wang to guarantee accurate results and optimal performance.[12] Especially when simulating critical systems, inferior PRNGs can lead to incorrect outcomes.[13] When it comes to first-principle calculations and molecular dynamics simulations, which make use of the diffusion and variational quantum Monte Carlo methods, most PRNGs can provide correct results under one sigma error bar, except for some underperforming PRNGs like RANDLUX and the short-period linear congruential generator (LCG).[9,14,15] Meanwhile, the autocorrelation length and CPU time can be influenced by the choice of PRNG.[8] However, such analysis of PRNGs on the world-line QMC has never been considered, so becomes necessary.

In this paper, we test the performance of different types of PRNG on the simulation of SSE methods. A quantity called QMC efficiency is introduced to evaluate the performance quantitatively. After checking different variables in one- and two-dimensional Heisenberg models, we provide a table explicitly demonstrating that the LCG is the best. The paper is organized as follows: in Section 2, we briefly review the QMC-SSE algorithm and define the efficiency of QMC;

†Corresponding author. E-mail: zhangxf@cqu.edu.cn

© 2024 Chinese Physical Society and IOP Publishing Ltd

in Section 3, we discuss the chosen PRNGs and the environment parameters of the computing platforms; in Section 4, the benchmarks of PRNGs are presented; in Section 5, we draw conclusions.

2. QMC-SSE method

For the quantum spin system, the Hamiltonian is usually constructed with diagonal and off-diagonal operators. Taking the Heisenberg model $H = J\sum_{\langle i,j\rangle}(S_i^x S_j^x + S_i^y S_j^y + S_i^z S_j^z)$ as an example, the diagonal part is the Ising term $S_i^z S_j^z$ and the off-diagonal one is the XY term $S_i^x S_j^x + S_i^y S_j^y = \frac{1}{2}(S_i^+ S_i^- +$ h.c.), which can exchange the spin configurations between the nearest-neighbor sites. Then, the series expansion of the partition function can be rewritten as

$$Z = \sum_{\alpha}\sum_{n=0}^{\infty}\sum_{S_n}\frac{(-\beta)^n}{n!}\langle\alpha|\prod_{i=1}^{n}H_{a_i,b_i}|\alpha\rangle, \tag{2}$$

where a_i and b_i label the type and position of the operators, and S_n represents the operator sequences. The program of SSE includes three steps: initialization, thermalization, and measurement.

In the initialization step, the parameters of the PRNG are set, especially the seed, so that the results are reproducible. Meanwhile, the information of the lattice is constructed, such as the position of the bonds, the coordinates of the sites, the relative positions of different sites, and so on. The configuration of the operator sequence and the status of each spin are also initialized. Most importantly, the weights of the operators are calculated, as well as the corresponding possibility of transferring one operator to the others. This is directly related to the updating of the operator sequence. In the computer program, the initialization step is only executed once, so the consideration of its efficiency becomes unnecessary.

The configuration of the operator sequence living in $d+1$ dimensions is updated in both thermalization and measurement steps. The updating algorithms are different,[16–18] and they aim to produce independent samples more efficiently. In the thermalization step, the system is approaching the ground state during the updating. It can be taken as thermal annealing because the number of operators increases during the updating, which is equivalent to decreasing the temperature. The thermalization step should follow a single Markov chain, so it cannot be parallelized.

In comparison, after reaching the ground state, the operator sequence can be distributed to several computing cores so that the measurement part can be parallelized and speeded up. Meanwhile, all the physical observables are calculated in each step, and their mean value $\langle O\rangle = \sum_\alpha \langle\alpha|O\exp(-\beta H)|\alpha\rangle/Z$ can be obtained by taking the Monte Carlo average $\langle O\rangle = \sum_i O_i/N_{\rm MC}$ where O_i is the value of observable at the ith step and the total number of measuring steps is $N_{\rm MC}$. The standard error can be estimated by using the bootstrap or jack-knife method.[19] Here, we consider the total energy $E = \langle H\rangle$, the total magnetization $M = \langle\sum_i S_i^z\rangle$ and its square $M^2 = \langle(\sum_i S_i^z)^2\rangle$, the structure factor $S(\boldsymbol{Q}) = \langle\sum_{l,l'} S_l^z S_{l'}^z \exp(-\mathrm{i}\boldsymbol{Q}\cdot(\boldsymbol{r}_l - \boldsymbol{r}_{l'}))\rangle/N^2$ and the winding number W^2, which is related to the spin stiffness (or superfluid density in bosonic language).[20]

The ideal updating process makes the nearest Monte Carlo steps uncorrelated. However, in the real simulation, this is extremely hard to achieve for all the observables. To quantify the correlation between Monte Carlo steps, the autocorrelation function is introduced as follows:[19]

$$C(t) = \langle O_i O_{i+t}\rangle - \langle O_i\rangle\langle O_{i+t}\rangle, \tag{3}$$

where t labels the Monte Carlo steps and is sometimes called Monte Carlo time. Then, the normalized autocorrelation function can be defined as $\Gamma(t) = C(t)/C(0)$. Typically, the autocorrelation function is an exponential decay $\exp(-t/\tau)$ at large time, so we can take τ as the autocorrelation time. This indicates the correlation between Monte Carlo steps will decay down to $1/\mathrm{e}$ after τ steps updates, so that O_i and $O_{i+\tau}$ can be approximately taken as Markov process or uncorrelated. However, there is often more than one decay mode. Therefore, here we use the integrated autocorrelation time defined as $\tau_{\rm int} = (1/2) + \sum_{t=1}^{n}\Gamma(t)$.[19]

The standard error of the physical observables can be reduced by increasing the number of measurement steps. However, the long-range correlation between Monte Carlo steps can weaken the accuracy of the Monte Carlo average. Thus, we introduce the number of effective steps $N_{\rm eff} = N_{\rm MC}/(2\tau_{\rm int})$ where the prefactor 2 is used to guarantee $N_{\rm eff} < N_{\rm MC}$. The computation time per step is $\bar{T}^{\rm C} = T_t^{\rm C}/N_{\rm MC}$, where $T_{\rm t}^{\rm C}$ is the total computation time. Then we name the effective steps per second as the QMC efficiency, which is defined as

$$\eta = \frac{N_{\rm eff}}{T_{\rm t}^{\rm C}} = \frac{1}{2\tau_{\rm int}\bar{T}^{\rm C}}. \tag{4}$$

The inverse of η is equal to the computation time per effective step, and we clearly find that a shorter autocorrelation time or computation time per step can give the QMC algorithm higher performance. This quantity is definitely different while considering different observables, because their autocorrelation times are determined by different modes of updating process.

3. PRNG and platform

The simplest but also fastest algorithm of PRNG is LCG,[21] which makes use of the linear congruence function as the iteration function $s_{i+1} = f(s_i) = \mathrm{Mod}(as_i + b, p)$ with all coefficients being positive integer. It is obvious that the maximum period is p, and all the coefficients a, b, and p should be

set within certain conditions so that the maximum period p can be reached.[22] Meanwhile, if the period is set to be the maximum of the 64-bit integer, the LCG algorithm can be strongly boosted by using the integer overflow. The integer overflow means the integer will throw the higher digits when the result is outside of the range after the operation, and it is equivalent to the function Mod( ). Although it is the fastest, the limitation of the digits of the register causes a serious problem that the period of LCG cannot exceed $2^{64}-1$.

In 1998, Matsumoto and Nishimura broke this constraint and invented a random number generator called the Mersenne twister (MT), which is based on the matrix linear congruential method.[10,23] The most popular version is the MT-19937, which has a very long period of $2^{19937}-1$. Then, Saito and Matsumoto introduced a new version of MT named the SIMD-oriented fast Mersenne twister (SFMT), which is twice as fast as conventional MT, and its period is extended to be incredibly large $2^{216091}-1$.[24] On the other hand, another new version of the MT method named the well equidistributed long-period linear generator (WELL) has better equal distribution and longer periods, and lower CPU time consumption.[25] The permuted congruential generator (PCG) was invented by O'Neill in 2014 and can provide a larger period with a smaller size register.[26] PCG can be taken as an improvement of the Xorshift method, which uses the shift and xor register method as the transform function.[26] Furthermore, we also consider another famous PRNG named KISS (after the "keep it simple, stupid" principle), which also has a long period 2^{123}.[27]

The comparison of the PRNGs has to be implemented in the same computing environment. The CPU chosen is the Intel Xeon Gold 6420R processor (dual-channel, 2.4 GHz, 24 cores, and 35.75 MB high-speed L3 Intel Smart Cache), and the total memory is 256 GB (16×16 GB DDR4). The operating system is Linux CentOS 7.6.1810 and the program compiler is GCC 4.8.5. The optimization flags of the compiler are set as -O3 and -mAVX. The SSE algorithm is the direct loop method[6] and is written in C++ programming language without GPU boosting. The simulated system sizes of the one- and two-dimensional Heisenberg models are set to be 500 and 24×24 under periodical boundary conditions, respectively. The inverse temperature is $\beta=1/T=10$ ($\beta=50$ for calculating the winding numbers). As the free parameter of SSE, the energy shift of the diagonal vertex is set to 0.3. The number of measurement steps is as large as 10^5 and the number of thermalization steps is half of that.

The Xeon processor utilizes the turbo boost technique, which will change the clock frequency of each processor. To rule out its influence, we use the clock( ) function in the ⟨time.h⟩ C library to count the system time and take an average of 1000 QMC runs (100 QMC runs for the two-dimensional case) with different initial seeds.

4. Results

First, the selection of the PRNGs definitely should not cause the QMC-SSE program to produce incorrect results. As mentioned before, some PRNGs (e.g., LCG) suffer from the "lattice" structure, and they may introduce serious problems, especially in the field of cryptography. Therefore, we examine various typical observables in one- and two-dimensional Heisenberg models and list them in Table 1. The largest deviation values are marked with colors. Their mean values are calculated by taking an average of many independent Markov chains with different initial seeds, and the standard errors with 2σ (95% confidence) are also shown. Here, the winding number is not considered in the one-dimensional Heisenberg model, because it is zero due to the absence of long-range off-diagonal order. From the numerical results, we can find there is no wrong result coming out with consideration of the statistical error. Thus, all the PRNGs listed are safe for QMC-SSE, including the LCG, which is commonly used. However, it appears the SFMT algorithm with a larger period may easily cause an apparently larger deviation.

The efficiency of the algorithm can be reflected by the autocorrelation time. As mentioned before, the updating of different variables is determined by different modes of the transfer matrix. Therefore, we list the integrated autocorrelation times of all the observables in Table 2. We find the autocorrelation times of the observables are strongly different and sensitive to different PRNGs, but the deviations are small, which means the selection of PRNGs cannot strongly affect the autocorrelation between Monte Carlo steps. However, we notice that the PCG and KISS demonstrate much smaller autocorrelation times of winding numbers with larger standard errors. The reason has not been determined yet, but we think it may be related to the topological properties of the winding number.

The main goal of the PRNGs is to produce independent samples with low computing resource costs. It is common to take MT-19937 or LCG as the PRNG when writing the QMC code. In recent decades, PRNGs have been continuously under development and their performance has strongly improved; for instance, the SFMT is twice as fast as the MT series as mentioned on the homepage of SFMT. Contrary to the usual ways of thinking, the periods of PRNGs have less effect on the computation time, so a shorter period does not mean faster. In the QMC algorithm, the PRNGs only participate in the updating process and are not involved in the measurement. Thus, the computation time is estimated by running only diagonal and loop updates after the thermalization steps. In other words, the computation time affected by the PRNGs is not related to the observables. In Fig. 1, we find that the influence of PRNGs is very serious in both dimensions. Counterintuitively, the SFMT with a smaller period costs a lot, especially in a two-dimensional system. In contrast, SFMT-11213, 19973, and

86243 show very good performance. The LCG is widely used in SSE by Sandvik, and it is extremely good. KISS is also a notable alternative choice, and the computation cost of WELL series PRNGs is still fine.

Finally, we can calculate the QMC efficiency of all the observables listed in Table 3. Then, to simplify the qualification of the PRNGs, we define the benchmark score as follows:

$$\mathrm{SC} = \frac{1}{n}\sum_{i=1}^{n}\frac{\eta_i}{\eta_i^{\mathrm{M}}}, \tag{5}$$

where η_i is the QMC efficiency for the i-th observable with η_i^{M} as the largest value among all the PRNGs, and n is the number of observables considered (e.g., $n = 4$ in 1D). Then, in one dimension, LCG, SFMT-11213, 19937 and 86243 are the best choices. In comparison, LCG and KISS perform excellently in two dimensions. After taking an average of both dimensions, we recommend the LCG as the best PRNG in QMC-SSE.

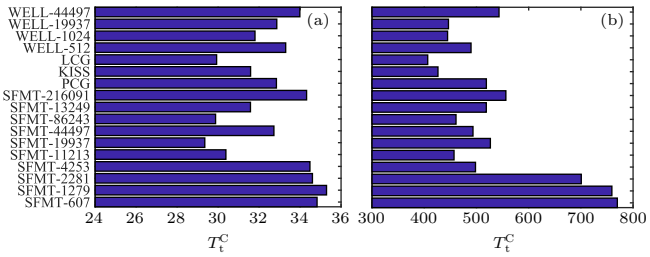


Fig. 1. The total computation time consumed for the (a) one–dimensional and (b) two-dimensional Heisenberg model within 10^5 Monte Carlo steps after thermalization.

Table 1. The numerical results of the one- and two-dimensional Heisenberg models calculated by the QMC-SSE method. The largest and smallest values are highlighted in red and green, respectively.

PRNG	One dimension				Two dimensions				
	E	M (10^{-2})	M^2	$S(\boldsymbol{Q})$ (10^{-3})	E	M (10^{-3})	M^2	$S(\boldsymbol{Q})$ (10^{-2})	W^2
SFMT-607	344.843(4)	−0.02(12)	5.713(3)	2.258(1)	673.670(3)	0.02(29)	0.4441(3)	4.076(1)	6.326(4)
SFMT-1279	344.845(4)	−0.03(12)	5.710(3)	2.259(1)	673.674(3)	−0.05(28)	0.4443(3)	4.074(1)	6.325(3)
SFMT-2281	344.840(4)	0.02(12)	5.714(3)	2.258(1)	673.674(3)	0.25(31)	0.4445(3)	4.073(1)	6.321(4)
SFMT-4253	344.843(4)	0.04(12)	5.713(3)	2.259(1)	673.675(3)	−0.02(29)	0.4449(3)	4.076(1)	6.322(4)
SFMT-11213	344.840(4)	−0.02(12)	5.713(3)	2.259(1)	673.672(3)	−0.31(32)	0.4444(3)	4.073(1)	6.316(4)
SFMT-19937	344.841(4)	0.05(11)	5.710(3)	2.259(1)	673.675(3)	0.37(31)	0.4445(3)	4.075(1)	6.325(4)
SFMT-44497	344.842(4)	−0.03(12)	5.711(3)	2.258(1)	673.669(3)	−0.11(28)	0.4447(3)	4.074(1)	6.320(4)
SFMT-86243	344.843(4)	0.01(11)	5.707(3)	2.260(1)	673.676(3)	−0.18(31)	0.4442(3)	4.077(1)	6.314(4)
SFMT-13249	344.839(4)	0.01(12)	5.710(3)	2.258(1)	673.677(3)	0.13(32)	0.4443(3)	4.074(1)	6.321(4)
SFMT-216091	344.843(4)	0.06(12)	5.714(3)	2.259(1)	673.670(3)	0.18(26)	0.4440(3)	4.073(1)	6.329(4)
PCG	344.840(4)	0.12(11)	5.710(3)	2.258(1)	673.671(3)	0.24(28)	0.4441(3)	4.072(1)	6.319(3)
KISS	344.843(4)	−0.05(12)	5.713(3)	2.260(1)	673.670(3)	−0.51(30)	0.4438(3)	4.072(1)	6.322(4)
LCG	344.842(4)	−0.04(12)	5.714(3)	2.259(1)	673.673(3)	−0.18(27)	0.4441(3)	4.072(1)	6.315(4)
WELL-512	344.842(4)	0.02(12)	5.712(3)	2.258(1)	673.673(3)	0.02(27)	0.4444(2)	4.074(1)	6.322(4)
WELL-1024	344.844(4)	0.12(11)	5.711(3)	2.259(1)	673.674(3)	0.01(26)	0.4449(3)	4.072(1)	6.322(4)
WELL-19937	344.842(4)	0.08(12)	5.711(3)	2.258(1)	673.678(3)	−0.26(31)	0.4450(3)	4.073(1)	6.325(4)
WELL-44497	344.842(4)	−0.04(12)	5.713(3)	2.258(1)	673.670(3)	−0.26(30)	0.4445(3)	4.074(1)	6.319(4)

Table 2. The autocorrelation time of different variables of the one- and two-dimensional Heisenberg model calculated by the QMC-SSE method. The largest and smallest values are highlighted in red and green, respectively.

PRNG	One dimension				Two dimensions				
	E	M	M^2	$S(\boldsymbol{Q})$	E	M	M^2	$S(\boldsymbol{Q})$	W^2
SFMT-607	4.276(15)	3.121(6)	1.600(3)	1.656(4)	2.204(7)	0.943(2)	0.647(1)	0.4150(3)	0.730(3)
SFMT-1279	4.274(14)	3.122(6)	1.597(3)	1.660(4)	2.197(7)	0.943(2)	0.648(1)	0.4148(3)	0.730(3)
SFMT-2281	4.264(14)	3.125(6)	1.601(3)	1.657(4)	2.213(7)	0.946(2)	0.649(1)	0.4150(3)	0.731(3)
SFMT-4253	4.262(14)	3.123(6)	1.598(3)	1.657(4)	2.208(7)	0.945(2)	0.647(1)	0.4151(3)	0.731(3)
SFMT-11213	4.290(14)	3.123(7)	1.596(3)	1.658(4)	2.195(7)	0.944(2)	0.648(1)	0.4153(3)	0.731(3)
SFMT-19937	4.270(14)	3.123(6)	1.599(3)	1.657(4)	2.206(7)	0.945(2)	0.646(1)	0.4145(3)	0.730(3)
SFMT-44497	4.275(14)	3.118(6)	1.599(3)	1.658(4)	2.200(7)	0.942(2)	0.647(1)	0.4152(3)	0.731(4)
SFMT-86243	4.270(14)	3.121(6)	1.597(3)	1.657(4)	2.238(9)	0.943(2)	0.648(1)	0.4150(3)	0.730(3)
SFMT-13249	4.267(14)	3.123(6)	1.598(3)	1.659(4)	2.217(8)	0.947(2)	0.649(1)	0.4155(3)	0.730(3)
SFMT-216091	4.269(14)	3.122(6)	1.600(3)	1.656(4)	2.212(8)	0.944(1)	0.650(1)	0.4150(3)	0.731(3)
PCG	4.272(14)	3.121(6)	1.598(3)	1.658(4)	2.192(6)	0.943(1)	0.647(1)	0.4154(3)	0.598(31)
KISS	4.272(14)	3.124(7)	1.597(3)	1.657(4)	2.202(8)	0.941(1)	0.647(1)	0.4150(3)	0.582(32)
LCG	4.265(14)	3.122(7)	1.595(3)	1.657(4)	2.206(8)	0.943(2)	0.645(1)	0.4151(3)	0.731(3)
WELL-512	4.263(14)	3.120(6)	1.601(3)	1.656(4)	2.202(7)	0.944(2)	0.648(1)	0.4147(3)	0.729(2)
WELL-1024	4.267(13)	3.124(6)	1.599(3)	1.659(4)	2.207(7)	0.943(1)	0.646(1)	0.4149(3)	0.731(2)
WELL-19937	4.285(14)	3.118(6)	1.598(3)	1.657(4)	2.195(7)	0.944(2)	0.649(1)	0.4151(3)	0.732(3)
WELL-44497	4.272(14)	3.123(6)	1.600(3)	1.655(4)	2.211(7)	0.942(2)	0.648(1)	0.4146(3)	0.732(3)

Table 3. The QMC efficiency and the scores of the QMC-SSE method. The recommended PRNGs are highlighted in red.

PRNG	One dimension					Two dimensions						Total
	E	M	M^2	$S(\boldsymbol{Q})$	Score	E	M	M^2	$S(\boldsymbol{Q})$	W^2	Score	score
SFMT-607	671	920	1794	1734	84	29	69	100	156	89	51	68
SFMT-1279	663	908	1775	1707	83	30	70	102	159	90	52	67
SFMT-2281	678	925	1805	1744	85	32	75	110	172	98	56	70
SFMT-4253	681	929	1815	1750	85	45	106	155	242	137	79	82
SFMT-11213	767	1054	2062	1986	97	50	116	169	263	150	86	91
SFMT-19937	798	1091	2130	2056	100	43	101	147	229	130	75	87
SFMT-44497	715	980	1911	1843	90	46	108	157	244	139	80	85
SFMT-86243	784	1073	2096	2020	98	48	115	168	261	149	85	92
SFMT-13249	742	1014	1982	1908	93	43	102	149	232	132	76	84
SFMT-216091	683	933	1822	1760	86	41	95	138	217	123	71	78
PCG	713	975	1905	1836	89	44	102	149	232	161	79	84
KISS	741	1014	1983	1910	93	53	125	181	283	201	96	95
LCG	784	1070	2095	2016	98	56	130	191	296	168	97	97
WELL-512	705	963	1876	1814	88	46	108	158	246	140	80	84
WELL-1024	737	1007	1966	1896	92	51	119	174	271	154	88	90
WELL-19937	710	976	1905	1837	89	51	119	172	270	153	88	89
WELL-44497	689	942	1838	1777	86	42	98	142	222	126	72	79

5. Conclusion

The performance of PRNGs in the QMC-SSE method is systematically analyzed in this work. The correctness of the algorithm is not ruined by the drawbacks of PRNGs, e.g., short period, lattice structure, and so on. Meanwhile, the autocorrelation time is also less sensitive to the choice of PRNG, except for the winding number while using LCG and KISS. We find that the computation time contributes to the major impact on the performance. After introducing the QMC efficiency and benchmark score, we provide a strong recommendation for the LCG as the best PRNG in the QMC-SSE method. If the lattice structure is still a worry, KISS and SFMT-86243 would be alternative solutions. The selection of the best PRNG may be highly relevant to the type of QMC, but the process of analysis we have introduced here can also be borrowed as a standard procedure.

Program availability

The code used in this article has been published on GitHub at https://github.com/LiuDongXu-01/QMC_RNG and is also openly available in Science Data Bank at https://doi.org/10.57760/sciencedb.j00113.00206. Download and decompress the code file QMC.tar.gz. Then, go into the decompressed folder. The readme file provides all the details on installing the program and how to use it.

Acknowledgments

Project supported by the National Natural Science Foundation of China (Grant Nos. 12274046, 11874094, and 12147102), Chongqing Natural Science Foundation (Grant No. CSTB2022NSCQ-JQX0018), and Fundamental Research Funds for the Central Universities (Grant No. 2021CDJZYJH-003).

References

[1] Hastings W K 1970 *Biometrika* **57** 97
[2] Troyer M and Werner P 2009 *AIP Conference Proceedings* **1162** 98
[3] Sandvik A W 1992 *J. Phys. A: Math. Gen.* **25** 3667
[4] Sandvik A W, Singh R R P and Campbell D K 1997 *Phys. Rev. B* **56** 14510
[5] Sandvik A W 1999 *Phys. Rev. B* **59** R14157
[6] Syljuåsen O F and Sandvik A W 2002 *Phys. Rev. E* **66** 046701
[7] Ferrenberg A M, Landau D P and Wong Y J 1992 *Phys. Rev. Lett.* **69** 3382
[8] Hongo K, Maezono R and Miura K 2010 *Journal of Computational Chemistry* **31** 2186
[9] Click T H, Liu A and Kaminski G A 2011 *Journal of Computational Chemistry* **32** 513
[10] L'Ecuyer P 1990 *Communications of the ACM* **33** 85
[11] Ripley B D and Cox D R 1983 *Proc. R. Soc. Lond. A* **389** 197
[12] Coddington P D 1996 *International Journal of Modern Physics C* **07** 295
[13] Filk T, Marcu M and Fredenhagen K 1985 *Phys. Lett. B* **165** 125
[14] Lüscher M 1994 *Comput. Phys. Commun.* **79** 100
[15] Marsaglia G and Zaman A 1991 *The Annals of Applied Probability* **1** 462
[16] Shao H, Guo W and Sandvik A W 2016 *Science* **352** 213
[17] Yan Z, Wu Y, Liu C, Syljuåsen O F, Lou J and Chen Y 2019 *Phys. Rev. B* **99** 165135
[18] Yan Z 2022 *Phys. Rev. B* **105** 184432
[19] Evertz H G 2003 *Adv. Phys.* **52** 1
[20] Pollock E L and Ceperley D M 1987 *Phys. Rev. B* **36** 8343
[21] Hull T E and Dobell A R 1962 *SIAM Review* **4** 230
[22] Knuth D E 2002 *The Art of Computer Programing: Seminumerical Algorithms*, Vol. 2 (Addison-Wesley, 2002)
[23] Matsumoto M and Nishimura T 1998 *ACM Trans. Model. Comput. Simul.* **8** 3
[24] Saito M and Matsumoto M 2008 *Monte Carlo and Quasi-Monte Carlo Methods* 2006, pp. 607–622
[25] Panneton F, L'Ecuyer P and Matsumoto M 2006 *ACM Trans. Math. Softw.* **32** 1
[26] O'Neill M E 2014 *PCG: A Family of Simple Fast Space-Efficient Statistically Good Algorithms for Random Number Generation*
[27] Rose G G 2018 *Cryptography Commun.* **10** 123