Derandomised lattice rules for high dimensional integration

Y. Kazashi[1] F. Y. Kuo[2] I. H. Sloan[3]

(Received 12 March 2019; revised 22 October 2019)

Abstract

We seek shifted lattice rules that are good for high dimensional integration over the unit cube in the setting of an unanchored weighted Sobolev space of functions with square-integrable mixed first derivatives. Many existing studies rely on random shifting of the lattice, whereas here we work with lattice rules with a deterministic shift. Specifically, we consider 'half-shifted' rules in which each component of the shift is an odd multiple of $1/(2N)$ where N is the number of points in the lattice. By applying the principle that *there is always at least one choice as good as the average*, we show that for a given generating vector there exists a half-shifted rule whose squared worst-case error differs from the shift-averaged squared worst-case error by a term of only order $1/N^2$. We carry out numerical experiments where the generating vector is chosen component-by-component (CBC), as for

DOI:10.21914/anziamj.v60i0.14110 gives this article, Published November 17, 2019, as part of the Proceedings of the 18th Biennial Computational Techniques and Applications Conference. ISSN 1445-8810.

randomly shifted lattices, and where the shift is chosen by a new 'CBC for shift' algorithm. The numerical results are encouraging.

Contents

1 Introduction **C248**
- 1.1 Function spaces and worst-case errors C250
- 1.2 Component-by-component constructions C251

2 Error analysis **C253**

3 Component-by-component for shift algorithm **C255**

4 Numerical results **C255**

1 Introduction

Lattice rules are often used for high dimensional integration over the unit cube, that is, for the numerical evaluation of the s-dimensional integral

$$I_s(f) := \int_0^1 \cdots \int_0^1 f(x_1, \dots, x_s)\, dx_1 \cdots dx_s = \int_{[0,1]^s} f(\boldsymbol{x})\, d\boldsymbol{x}\,. \tag{1}$$

A *shifted lattice rule* for the approximation of the integral is an equal weight cubature rule of the form

$$Q_{N,s}(\boldsymbol{z}, \boldsymbol{\Delta}; f) := \frac{1}{N} \sum_{k=1}^{N} f\left(\left\{\frac{k\boldsymbol{z}}{N} + \boldsymbol{\Delta}\right\}\right), \tag{2}$$

where $\boldsymbol{z} \in \{1, \dots, N-1\}^s$ is the *generating vector*, $\boldsymbol{\Delta} \in [0, 1]^s$ is the *shift*, while the braces around an s-vector indicate that each component of the vector is to be replaced by its fractional part in $[0, 1)$. The special case $\boldsymbol{\Delta} = \boldsymbol{0}$ yields the unshifted lattice rule which has been proved to work well for periodic functions [6]. If the integrand is not periodic, then the shift plays a useful role. The implementation of a shifted lattice rule is relatively easy once the vectors $\boldsymbol{z}$ and $\boldsymbol{\Delta}$ are prescribed, even when s is very large, say, in the tens of thousands.

The central concern of this article is the construction of a good shift vector $\boldsymbol{\Delta}$, given a specific choice of a good $\boldsymbol{z}$. At the present time the overwhelmingly favoured method for fixing the shifts in a non-periodic setting is to choose them *randomly*. In a *randomly shifted lattice rule* the shift $\boldsymbol{\Delta}$ is chosen from a uniform distribution on $[0,1]^s$, and the integral (1) is approximated by an empirical estimate of the expected value $\frac{1}{q}\sum_{i=1}^{q} Q_{N,s}(\boldsymbol{z},\boldsymbol{\Delta}_i\,;\,f)$, where q is some fixed number and $\boldsymbol{\Delta}_1,\ldots,\boldsymbol{\Delta}_q$ are q independent samples from the uniform distribution on $[0,1]^s$. With the shift chosen randomly, all that remains in the randomly shifted case is to construct the integer vector $\boldsymbol{z}$, which is done very effectively by using the *component-by-component* (CBC) construction to yield a vector $\boldsymbol{z}^*$ that gives a satisfactorily small value of the *shift-averaged worst-case error* [1].

In the present article we construct a new kind of shifted lattice rule which is *derandomised* in the sense that the generating vector is the same $\boldsymbol{z}^*$ determined by the CBC algorithm for the shift-averaged worst-case error, while the shift $\boldsymbol{\Delta}^*$ is determined by a new CBC construction, 'CBC *for shift*': the components of the shift vector are obtained one at a time, chosen from the *odd multiples of* $1/(2N)$. We argue that there is a significant potential cost saving in this deterministic alternative, in that it becomes no longer necessary to compute an empirical average over shifts.

Approaches to estimating the error for lattice rules for non-periodic functions without randomisation include those of Dick et al. [2] and Goda et al. [3], where a mapping called the *tent transform* is applied to the lattice rule. However, in this article, no transformation of the lattice points is considered.

1.1 Function spaces and worst-case errors

The central element in any CBC construction is the *worst-case error* which, for the case of the shifted lattice rule (2) and a Hilbert space H_s, is defined by

$$e_{N,s}(\boldsymbol{z},\boldsymbol{\Delta}) := \sup_{f\in H_s\,,\,\|f\|_{H_s}\leqslant 1} |Q_{N,s}(\boldsymbol{z},\boldsymbol{\Delta};f) - I_s(f)|\,.$$

Here we consider a weighted *unanchored* Sobolev space of functions with square-integrable mixed first derivatives on $(0,1)^s$ and squared norm

$$\|f\|^2_{H_{s,\gamma}} := \sum_{\mathfrak{u}\subseteq\{1:s\}} \gamma_{\mathfrak{u}}^{-1} \int_{[0,1]^{|\mathfrak{u}|}} \left(\int_{[0,1]^{s-|\mathfrak{u}|}} \frac{\partial^{|\mathfrak{u}|} f}{\partial \boldsymbol{x}_{\mathfrak{u}}}(\boldsymbol{x}_{\mathfrak{u}}\,;\boldsymbol{x}_{\{1:s\}\setminus\mathfrak{u}})\, d\boldsymbol{x}_{\{1:s\}\setminus\mathfrak{u}} \right)^2 d\boldsymbol{x}_{\mathfrak{u}}\,,$$

where $\{1:s\} = \{1,2,\ldots,s\}$, $\boldsymbol{\gamma}_{\mathfrak{u}}$ is a positive number which is the 'weight' corresponding to the subset $\mathfrak{u}\subseteq\{1:s\}$ with $\boldsymbol{\gamma}_\emptyset = 1$, and $\boldsymbol{x}_{\mathfrak{u}}$ denotes the variables x_j for $j\in\mathfrak{u}$. Suitably decaying weights are essential if we are to have error bounds independent of dimension [8]. The squared worst-case error has an explicit formula [e.g., 1]

$$e^2_{N,s}(\boldsymbol{z},\boldsymbol{\Delta}) = \frac{1}{N^2}\sum_{k=1}^{N}\sum_{k'=1}^{N}\sum_{\emptyset\neq\mathfrak{u}\subseteq\{1:s\}} \gamma_{\mathfrak{u}} \prod_{j\in\mathfrak{u}} \left[\frac{1}{2}B_2\left(\left\{\frac{(k-k')z_j}{N}\right\}\right) + A_{k,k',z_j}(\Delta_j)\right], \tag{3}$$

where $B_2(x) = x^2 - x + 1/6$ for $x\in[0,1]$ denotes the Bernoulli polynomial of degree two and

$$A_{k,k',z}(\Delta) := \left(\left\{\frac{kz}{N}+\Delta\right\}-\frac{1}{2}\right)\left(\left\{\frac{k'z}{N}+\Delta\right\}-\frac{1}{2}\right).$$

For the randomly shifted lattice rule the relevant form of the worst-case error is the *shift-averaged worst-case error* $e^{sh}_{N,s}(\boldsymbol{z})$ defined by

$$[e^{sh}_{N,s}(\boldsymbol{z})]^2 := \int_{[0,1]^s} e^2_{N,s}(\boldsymbol{z},\boldsymbol{\Delta})\, d\boldsymbol{\Delta} = \frac{1}{N}\sum_{k=1}^{N}\sum_{\emptyset\neq\mathfrak{u}\subseteq\{1:s\}} \gamma_{\mathfrak{u}} \prod_{j\in\mathfrak{u}} B_2\left(\left\{\frac{kz_j}{N}\right\}\right), \tag{4}$$

and $[e^{\mathrm{sh}}_{N,s}(\boldsymbol{z})]^2$ is precisely the expected value of the squared worst-case error taken with respect to the random shift. The double sum over k, k' in (3) simplified to a single sum over k in (4).

1.2 Component-by-component constructions

The principle of a CBC construction is that, at stage j, one determines the jth component of the cubature points by seeking to minimise an error criterion for the j-dimensional problem; then, with that component fixed, one moves on to the next component, never going back.

In the case of randomly shifted lattice rules, we first choose $z_1^* = 1$, and then, for $j = 1, 2, \ldots, s-1$, once $z_1^*, z_2^*, \ldots, z_j^*$ are fixed, z_{j+1} is chosen to be the element from $\{1, \ldots, N-1\}$ that gives the smallest error $[e^{\mathrm{sh}}_{N,j+1}(z_1^*, \ldots, z_j^*, z_{j+1})]^2$. The cost of the CBC algorithm for constructing $\boldsymbol{z}^*$ up to s dimensions is of order $s\, N \log N$ using fast Fourier transforms [5] for the simplest case of 'product weights' in which there is only one sequence of weight parameters $\gamma_1, \gamma_2, \ldots, \gamma_s$ and the value of $\gamma_{\mathfrak{u}}$ is taken to be the product $\prod_{j\in\mathfrak{u}} \gamma_j$. In this case the sum over $\mathfrak{u}$ in (4) can be rewritten as a product of s factors.

The proven quality of the CBC construction for randomly shifted lattice rules is very good in the sense that, with ζ the Riemann zeta function and φ the Euler totient function, for all $\lambda \in (\frac{1}{2}, 1]$ [e.g., 1],

$$e^{\mathrm{sh}}_{N,s}(\boldsymbol{z}^*) \leqslant \left[\frac{1}{\varphi(N)} \sum_{\emptyset \neq \mathfrak{u} \subseteq \{1:s\}} \gamma_{\mathfrak{u}}^{\lambda} \left(\frac{2\zeta(2\lambda)}{(2\pi^2)^{\lambda}}\right)^{|\mathfrak{u}|}\right]^{1/(2\lambda)} . \tag{5}$$

It follows from the definition (4) that for $f \in H_{s,\gamma}$ the error bound for the randomly shifted lattice rule constructed by CBC is

$$\sqrt{\mathbb{E}\left[|Q_{N,s}(\boldsymbol{z}^*, \cdot\,; f) - I_s(f)|^2\right]} \leqslant \left[\frac{1}{\varphi(N)} \sum_{\emptyset \neq \mathfrak{u} \subseteq \{1:s\}} \gamma_{\mathfrak{u}}^{\lambda} \left(\frac{2\zeta(2\lambda)}{(2\pi^2)^{\lambda}}\right)^{|\mathfrak{u}|}\right]^{1/(2\lambda)} \|f\|_{H_{s,\gamma}} .$$

When N is prime we have $\varphi(N) = N - 1$. Thus the convergence rate is arbitrarily close to $1/N$ as $\lambda \to 1/2$, but with a constant that blows up as $\lambda \to 1/2$ because $\zeta(2\lambda) \to \infty$.

For our new derandomised lattice rule we take the components of the generating vector to be $z_1^*, z_2^*, \ldots, z_s^*$, as determined by the CBC algorithm for randomly shifted lattice rules. We then determine the components of the shift by a new CBC for shift algorithm (see Section 3), in which at stage $j \geqslant 0$, with $\Delta_1^*, \ldots, \Delta_j^*$ already fixed, we choose Δ_{j+1} by minimising the squared worst-case error $e^2_{N,j+1}((z_1^*, \ldots, z_j^*, z_{j+1}^*), (\Delta_1^*, \ldots, \Delta_j^*, \Delta_{j+1}))$. Of course it is not possible to check all real numbers in $[0, 1)$ for desirable values of $\Delta_1, \ldots, \Delta_s$. We argue that it is sufficient to restrict the set of possible shift components to the odd multiples of $1/(2N)$, that is, to the N values $S_N := \{1/(2N), 3/(2N), \ldots, (2N-1)/(2N)\}$.

Theorem 1 presents our argument for the sufficiency of restricting the search over shifts to the odd multiples of $1/(2N)$. In this theorem we show that for any choice of generating vector $\boldsymbol{z}$, the average of the squared worst-case error over all shifts in $[0, 1]^s$ differs from the average over the discrete set S_N^s by a term of only order $1/N^2$.

The restriction from the continuous interval $[0, 1]$ to the discrete set S_N for the shift was previously considered by Sloan et al. [7] in a different CBC algorithm which constructs the components of $\boldsymbol{z}$ and $\boldsymbol{\Delta}$ simultaneously, in the order of $z_1, \Delta_1, z_2, \Delta_2, \ldots$.

Now we discuss the error with respect to the shift $\boldsymbol{\Delta}^*$ obtained by the present CBC for shift algorithm. Define the ratio

$$\kappa(N, s) := \frac{e_{N,s}(\boldsymbol{z}^*, \boldsymbol{\Delta}^*)}{e^{\mathrm{sh}}_{N,s}(\boldsymbol{z}^*)} . \tag{6}$$

Then, from the definition of the worst-case error and using (5) the error bound for the present CBC algorithm is

$$\begin{aligned}|Q_{N,s}(\boldsymbol{z}^*, \boldsymbol{\Delta}^*; f) - I_s(f)| &\leqslant \kappa(N,s)\, e^{\mathrm{sh}}_{N,s}(\boldsymbol{z}^*)\, \|f\|_{H_{s,\gamma}} \\ &\leqslant \kappa(N,s) \left[\frac{1}{\varphi(N)} \sum_{\emptyset \neq u \subseteq \{1:s\}} \gamma_u^\lambda \left(\frac{2\zeta(2\lambda)}{(2\pi^2)^\lambda} \right)^{|u|} \right]^{1/(2\lambda)} \|f\|_{H_{s,\gamma}} ,\end{aligned}$$

for all $\lambda \in (1/2, 1]$. This is an *explicit* and *deterministic* error bound in which in any practical situation $\kappa(N,s)$ is a known constant. Numerical experiments in Section 3 suggest that $\kappa(N,s)$ can often be smaller than one, making the derandomised option attractive in practice.

The presented CBC for shift algorithm is expensive: the cost of a single evaluation of the worst-case error (3) is of order sN^2 in the simplest case of product weights, and therefore the cost of a search over N values of the shift up to dimension s is of order sN^3 (if we store the products during the search). But the cost is an off-line cost, since spare computing capacity can be used to complement existing CBC vectors $\boldsymbol{z}^*$ for randomly shifted lattice rules by deterministic shifts $\boldsymbol{\Delta}^*$ generated by the CBC for shift algorithm.

2 Error analysis

Theorem 1 shows that for any choice of generating vector $\boldsymbol{z}$, the squared worst-case error with shift averaged over S_N^s, defined by

$$[e^{\frac{1}{2}\mathrm{sh}}_{N,s}(\boldsymbol{z})]^2 := \frac{1}{N^s} \sum_{\boldsymbol{\Delta} \in S_N^s} e^2_{N,s}(\boldsymbol{z}; \boldsymbol{\Delta}) , \tag{7}$$

differs from the average of the squared worst-case error over all shifts $[e^{\mathrm{sh}}_{N,s}(\boldsymbol{z})]^2$ by a term of only order $1/N^2$.

Theorem 1. *For arbitrary* $\boldsymbol{z} \in \{1, \ldots, N-1\}^s$ *, with* $e^{\mathrm{sh}}_{N,s}(\boldsymbol{z})$ *and* $e^{\frac{1}{2}\mathrm{sh}}_{N,s}(\boldsymbol{z})$ *as defined in* (4) *and* (7)*, respectively, we have*

$$\left|[e^{\mathrm{sh}}_{N,s}(\boldsymbol{z})]^2 - [e^{\frac{1}{2}\mathrm{sh}}_{N,s}(\boldsymbol{z})]^2\right| \leqslant \frac{1}{4N^2} \sum_{\emptyset \neq \mathfrak{u} \subseteq \{1:s\}} \gamma_{\mathfrak{u}} \left(\frac{1}{3}\right)^{|\mathfrak{u}|} |\mathfrak{u}| \,.$$

Proof: We see from (3) that

$$[e^{\mathrm{sh}}_{N,s}(\boldsymbol{z})]^2 - [e^{\frac{1}{2}\mathrm{sh}}_{N,s}(\boldsymbol{z})]^2 = \frac{1}{N^2} \sum_{k=1}^{N} \sum_{k'=1}^{N} \sum_{\emptyset \neq \mathfrak{u} \subseteq \{1:s\}} \gamma_{\mathfrak{u}} \left(\prod_{j \in \mathfrak{u}} a_j^{k,k'} - \prod_{j \in \mathfrak{u}} b_j^{k,k'} \right),$$

where for $k, k' = 1, \ldots, N$, $j = 1, \ldots, s$, $m = 1, \ldots, N$,

$$a_j^{k,k'} := c_j^{k,k'} + \int_0^1 A_{k,k',z_j}(\Delta)\, d\Delta\,, \qquad b_j^{k,k'} := c_j^{k,k'} + \frac{1}{N} \sum_{m=1}^{N} A_{k,k',z_j}(\mu_m)\,,$$

$$c_j^{k,k'} := \frac{1}{2} B_2 \left(\left\{ \frac{(k-k')z_j}{N} \right\} \right), \qquad \mu_m := \frac{2m-1}{2N}\,.$$

Since $|B_2(x)| \leqslant 1/6$ for all $x \in [0,1]$ and $|(x-1/2)(y-1/2)| \leqslant 1/4$ for all $x, y \in [0,1)$, we have trivially $|a_j^{k,k'}| \leqslant 1/3$ and $|b_j^{k,k'}| \leqslant 1/3$. It follows by induction that

$$\left| \prod_{j \in \mathfrak{u}} a_j^{k,k'} - \prod_{j \in \mathfrak{u}} b_j^{k,k'} \right| \leqslant \left(\frac{1}{3}\right)^{|\mathfrak{u}|-1} \sum_{j \in \mathfrak{u}} \left| a_j^{k,k'} - b_j^{k,k'} \right|.$$

We therefore consider the difference

$$\begin{aligned} a_j^{k,k'} - b_j^{k,k'} &= \int_0^1 A_{k,k',z_j}(\Delta)\, d\Delta - \frac{1}{N} \sum_{m=1}^{N} A_{k,k',z_j}(\mu_m) \\ &= \sum_{m=1}^{N} \left[\int_{(m-1)/N}^{m/N} A_{k,k',z_j}(\Delta)\, d\Delta - \frac{1}{N} A_{k,k',z_j}\left(\frac{2m-1}{2N}\right) \right], \end{aligned}$$

which is precisely the error of a *composite midpoint rule* approximation to the integral of

$$A_{k,k',z_j}(\Delta) = \left\{\frac{kz_j}{N} + \Delta\right\}\left\{\frac{k'z_j}{N} + \Delta\right\} - \frac{1}{2}\left\{\frac{kz_j}{N} + \Delta\right\} - \frac{1}{2}\left\{\frac{k'z_j}{N} + \Delta\right\} + \frac{1}{4}.$$

Since kz_j/N is a multiple of $1/N$, the function $\{kz_j/N + \Delta\}$ is *linear* in Δ on each subinterval $[(m-1)/N, m/N)$ of length $1/N$, and so the midpoint rule is exact on each subinterval. The same conclusion holds for $\{k'z_j/N+\Delta\}$. On the other hand, the expression $\{kz_j/N + \Delta\}\{k'z_j/N + \Delta\}$ as a function of Δ is *quadratic* on each subinterval $[(m-1)/N, m/N)$, and its second derivative is the constant function 2, which is uniformly continuous on $((m-1)/N, m/N)$ and can be uniquely extended to $[(m-1)/N, m/N]$. Therefore, the midpoint rule has error bounded by $1/(12N^3)$ on each subinterval, leading to the total error $|a_j^{k,k'} - b_j^{k,k'}| \leqslant 1/(12N^2)$, and in turn yielding

$$\left|\prod_{j\in u} a_j^{k,k'} - \prod_{j\in u} b_j^{k,k'}\right| \leqslant \left(\frac{1}{3}\right)^{|u|-1} \frac{|u|}{12N^2}.$$

This completes the proof. ♠

3 Component-by-component for shift algorithm

Theorem 1 provides a good motivation for Algorithm 1.

4 Numerical results

We ran the CBC for shift algorithm in weighted unanchored Sobolev spaces with product weights $\gamma_j = 1/j^2$, $\gamma_j = 0.9^j$, $\gamma_j = 0.75^j$, and $\gamma_j = 0.5^j$, with

Algorithm 1 CBC for shift

Input: $s_{\max}$, N, and $z_1^*, \dots z_{s_{\max}}^*$, a generating vector obtained by the CBC construction for randomly shifted lattice rules.

Output: shifts $\Delta_1^*, \dots, \Delta_{s_{\max}}^* \in S_N$, and

$$\kappa(N,s) = \frac{e_{N,s}((z_1^*,\dots,z_s^*),(\Delta_1^*,\dots,\Delta_s^*))}{e_{N,s}^{\mathrm{sh}}(z_1^*,\dots,z_s^*)}, \quad s = 1,\dots,s_{\max}.$$

Do

$$\Delta_1^* \in \operatorname{argmin}\left\{e_{N,1}^2(z_1^*,\Delta_1) \mid \Delta_1 \in S_N\right\},$$

and $\kappa(N,1) = e_{N,1}(z_1^*,\Delta_1^*)/e_{N,1}^{\mathrm{sh}}(z_1^*)$,

for s from 2 to $s_{\max}$ **do**

$$\Delta_s^* \in \operatorname{argmin}\left\{e_{N,s}^2((z_1^*,\dots,z_s^*),(\Delta_1^*,\dots,\Delta_{s-1}^*,\Delta_s)) \mid \Delta_s \in S_N\right\},$$

and $\kappa(N,s) = e_{N,s}((z_1^*,\dots,z_s^*),(\Delta_1^*,\dots,\Delta_s^*))/e_{N,s}^{\mathrm{sh}}(z_1^*,\dots,z_s^*)$,

end for

the number of points $N = 1024$ and 2048. We used the lattice generating vectors $\boldsymbol{z}^*$ as in the original version developed by Kuo [4].

Table 1 shows the values of the indices m_s^* for the components of the shifts $\Delta_s^* = (2m_s^* - 1)/(2N)$ together with the values of $\kappa(N,s)$, for the case $N = 2048$ and $\gamma_j = 1/j^2$. As a comparison, we also provide the values of the ratio (6) with $\boldsymbol{\Delta}^*$ replaced by the zero shift vector, denoting the new ratio by $\kappa_0(N,s)$. We see that $\kappa(N,s) < 1$, whereas $\kappa_0(N,s) > 1$.

Table 2 shows the same as Table 1, but for the case $\gamma_j = 0.5^j$. Again, we see that $\kappa(N,s) < 1$, whereas $\kappa_0(N,s) > 1$. The same observation holds for the other cases that we considered (not shown).

Acknowledgements We gratefully acknowledge the financial support from the Australian Research Council (DP180101356).

Table 1: Shifts $\Delta_s^* = (2m_s^* - 1)/(2N)$ and error ratio $\kappa(N, s)$ obtained by the CBC for shift algorithm for $N = 2048$ and weight $\gamma_j = 1/j^2$ for dimensions $s = 1, \ldots, 50$. Also tabulated is $\kappa_0(N, s)$, the value of $\kappa(N, s)$ corresponding to zero shift. We see that $\kappa(N, s) < 1$.

s	m_s^*	$\kappa(2048, s)$	$\kappa_0(2048, s)$	s	m_s^*	$\kappa(2048, s)$	$\kappa_0(2048, s)$
1	1	0.7082	1.4148	26	626	0.8686	1.1170
2	227	0.7748	1.2426	27	1987	0.8691	1.1162
3	17	0.8047	1.1841	28	1676	0.8696	1.1165
4	1955	0.8176	1.1599	29	1323	0.8698	1.1161
5	1273	0.8276	1.1642	30	1037	0.8702	1.1156
6	1250	0.8358	1.1532	31	416	0.8706	1.1161
7	1698	0.8414	1.1404	32	416	0.8706	1.1163
8	1970	0.8456	1.1357	33	928	0.8708	1.1161
9	476	0.8480	1.1342	34	928	0.8708	1.1161
10	646	0.8507	1.1304	35	711	0.8712	1.1157
11	779	0.8535	1.1293	36	711	0.8712	1.1153
12	1093	0.8558	1.1264	37	1852	0.8715	1.1152
13	1498	0.8572	1.1234	38	1852	0.8715	1.1155
14	550	0.8591	1.1223	39	785	0.8718	1.1151
15	1218	0.8603	1.1230	40	785	0.8718	1.1153
16	1124	0.8614	1.1214	41	696	0.8721	1.1151
17	135	0.8624	1.1206	42	1497	0.8758	1.1148
18	717	0.8635	1.1200	43	1587	0.8760	1.1146
19	854	0.8645	1.1192	44	638	0.8762	1.1145
20	1634	0.8652	1.1183	45	848	0.8764	1.1141
21	1692	0.8658	1.1178	46	954	0.8765	1.1139
22	1002	0.8665	1.1164	47	1042	0.8767	1.1136
23	1034	0.8670	1.1171	48	20	0.8769	1.1136
24	249	0.8675	1.1171	49	589	0.8770	1.1138
25	1477	0.8681	1.1163	50	617	0.8771	1.1138

Table 2: Shifts $\Delta_s^* = (2m_s^* - 1)/(2N)$ and error ratio $\kappa(N,s)$ obtained by the CBC for shift algorithm for $N = 2048$ and weight $\gamma_j = 0.5^j$ for dimensions $s = 1, \ldots, 50$. Also tabulated is $\kappa_0(N,s)$, the value of $\kappa(N,s)$ corresponding to zero shift. We see that $\kappa(N,s) < 1$.

s	m_s^*	$\kappa(2048,s)$	$\kappa_0(2048,s)$	s	m_s^*	$\kappa(2048,s)$	$\kappa_0(2048,s)$
1	1	0.7082	1.4148	26	11	0.8902	1.2372
2	227	0.7748	1.2426	27	1696	0.8970	1.2537
3	17	0.8047	1.1841	28	820	0.8965	1.2568
4	1955	0.8176	1.1599	29	1629	0.9005	1.2693
5	422	0.8291	1.1464	30	1272	0.9041	1.2799
6	1698	0.8363	1.1307	31	1661	0.9048	1.2830
7	1917	0.8418	1.1319	32	633	0.9091	1.2912
8	2005	0.8456	1.1271	33	205	0.9129	1.2986
9	5	0.8484	1.1214	34	1841	0.9162	1.3054
10	135	0.8518	1.1161	35	2038	0.9171	1.3075
11	1139	0.8539	1.1181	36	1433	0.9199	1.3130
12	1410	0.8571	1.1118	37	405	0.9204	1.3149
13	982	0.8593	1.1098	38	1042	0.9215	1.3170
14	1151	0.8605	1.1076	39	589	0.9224	1.3191
15	751	0.8621	1.1049	40	1068	0.9246	1.3229
16	1043	0.8636	1.1029	41	1763	0.9271	1.3263
17	1083	0.8648	1.1076	42	1364	0.9293	1.3295
18	412	0.8661	1.1071	43	1946	0.9314	1.3325
19	211	0.8671	1.1064	44	214	0.9320	1.3337
20	854	0.8679	1.1055	45	1511	0.9338	1.3362
21	418	0.8686	1.1367	46	1835	0.9344	1.3374
22	849	0.8692	1.1648	47	128	0.9359	1.3395
23	13	0.8769	1.1979	48	1500	0.9365	1.3405
24	1280	0.8771	1.1977	49	1023	0.9379	1.3424
25	1229	0.8825	1.2174	50	561	0.9391	1.3442

References

[1] J. Dick, F. Y. Kuo, and I. H. Sloan. “High-dimensional integration: The quasi-Monte Carlo way”. In: *Acta Numer.* 22 (2013), pp. 133–288. DOI: 10.1017/S0962492913000044 (cit. on pp. C249, C250, C251).

[2] J. Dick, D. Nuyens, and F. Pillichshammer. “Lattice rules for nonperiodic smooth integrands”. In: *Numer. Math.* 126.2 (2014), pp. 259–291. DOI: 10.1007/s00211-013-0566-0 (cit. on p. C249).

[3] T. Goda, K. Suzuki, and T. Yoshiki. “Lattice rules in non-periodic subspaces of Sobolev spaces”. In: *Numer. Math.* 141.2 (2019), pp. 399–427. DOI: 10.1007/s00211-018-1003-1 (cit. on p. C249).

[4] F. Y. Kuo. *Lattice rule generating vectors*. URL: http://web.maths.unsw.edu.au/~fkuo/lattice/index.html (visited on 02/27/2019) (cit. on p. C256).

[5] D. Nuyens and R. Cools. “Fast algorithms for component-by-component construction of rank-1 lattice rules in shift-invariant reproducing kernel Hilbert spaces”. In: *Math. Comput.* 75 (2006), pp. 903–920. DOI: 10.1090/S0025-5718-06-01785-6 (cit. on p. C251).

[6] I. H. Sloan and S. Joe. *Lattice methods for multiple integration*. Oxford Science Publications. Clarendon Press and Oxford University Press, 1994. URL: https://global.oup.com/academic/product/lattice-methods-for-multiple-integration-9780198534723 (cit. on p. C249).

[7] I. H. Sloan, F. Y. Kuo, and S. Joe. “On the step-by-step construction of quasi-Monte Carlo integration rules that achieve strong tractability error bounds in weighted Sobolev spaces”. In: *Math. Comput.* 71 (2002), pp. 1609–1641. DOI: 10.1090/S0025-5718-02-01420-5 (cit. on p. C252).

[8] I. H Sloan and H. Woźniakowski. “When are quasi-Monte Carlo algorithms efficient for high dimensional integrals?” In: *J. Complex.* 14.1 (1998), pp. 1–33. DOI: 10.1006/jcom.1997.0463 (cit. on p. C250).

Author addresses

1. **Y. Kazashi**, Mathematics Institute, CSQI, École Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland.
   mailto:yoshihito.kazashi@epfl.ch
2. **F. Y. Kuo**, School of Mathematics and Statistics, University of New South Wales, Sydney NSW 2052, Australia.
   mailto:f.kuo@unsw.edu.au
3. **I. H. Sloan**, School of Mathematics and Statistics, University of New South Wales, Sydney NSW 2052, Australia.
   mailto:i.sloan@unsw.edu.au