ANZIAM J. 54(2013), 216–219
doi:10.1017/S144618111300014X

CORRECTION TO "QUASI-MONTE CARLO METHODS FOR HIGH-DIMENSIONAL INTEGRATION: THE STANDARD (WEIGHTED HILBERT SPACE) SETTING AND BEYOND"

F. Y. KUO[1], CH. SCHWAB[2] and I. H. SLOAN[✉ 1]

Correction

We report an error in Theorem 4.1 of our paper [F. Y. Kuo, Ch. Schwab and I. H. Sloan, "Quasi Monte-Carlo methods for high-dimensional integration: The standard (weighted Hilbert space) setting and beyond", *ANZIAM J.* **53** (2011) 1–37]. This error does not affect the case of product weights, and is not relevant to the *unanchored* variant of the Sobolev space, but requires a modification of the CBC construction of lattice rules for the case of the *anchored* Sobolev space and general weights. We provide the necessary correction to Theorem 4.1, and to a related error in Theorem 4.3. The error bounds are unaffected by this correction, but the CBC construction in the anchored Sobolev space becomes more costly than in the unanchored case, and the resulting lattice rule is no longer extensible in dimension. For these reasons, we recommend the unanchored Sobolev space for the practical construction of lattice rules.

Equation (4.5) of our paper is invalid for the anchored Sobolev space ($m = a^2 - a + 1/3$ with a denoting the anchor) because the auxiliary weights defined by equation (4.3) implicitly depend on the dimension s. The unanchored case ($m = 0$) is unaffected because the auxiliary weights are the same as the original weights. We state a revised version of Theorem 4.1 and outline a correction to its proof. We also state a revised version of Theorem 4.3.

[1]School of Mathematics and Statistics, University of New South Wales, Sydney NSW 2052, Australia; e-mail: f.kuo@unsw.edu.au, i.sloan@unsw.edu.au.
[2]Seminar for Applied Mathematics, ETH Zürich, ETH Zentrum, HG G57.1, CH8092 Zürich, Switzerland; e-mail: christoph.schwab@sam.math.ethz.ch.

REVISED THEOREM 4.1. *A generating vector* $\boldsymbol{z}$ *can be constructed by a CBC algorithm such that, for any* $\lambda \in (1/2, 1]$,

$$e^2_{N,s,\boldsymbol{\gamma}}(\boldsymbol{z}) \le \Big(\sum_{\emptyset \ne \mathfrak{u} \subseteq \{1:s\}} \gamma_{\mathfrak{u}}^{\lambda} (\rho(\lambda))^{|\mathfrak{u}|} \Big)^{1/\lambda} [\phi(N)]^{-1/\lambda},$$

with

$$\rho(\lambda) := \frac{2\zeta(2\lambda)}{(2\pi^2)^{\lambda}} + m^{\lambda},$$

where $m = 0$ *in the unanchored case and* $m = a^2 - a + 1/3$ *in the anchored case with anchor* a, $\zeta(x)$ *is the Riemann zeta function as in Section* 1.5, *and* $\phi(N)$ *is the Euler totient function given by equation* (4.2).

(i) *For product weights* ($\gamma_{\mathfrak{u}} = \prod_{j \in \mathfrak{u}} \gamma_j$), *the CBC algorithm minimizes*

$$e^2_{N,d,\boldsymbol{\gamma}}(z_1, \ldots, z_d) = -\prod_{j=1}^{d} (1 + m\gamma_j) + \frac{1}{N} \sum_{k=0}^{N-1} \prod_{j=1}^{d} \Big(1 + \gamma_j \Big[B_2\Big(\Big\{\frac{kz_j}{N}\Big\}\Big) + m \Big]\Big),$$

step by step for each $d = 2, 3, \ldots, s$. *In this case, the CBC algorithm is extensible in* s.

(ii) *In the unanchored case with general nonproduct weights, the CBC algorithm minimizes*

$$e^2_{N,d,\boldsymbol{\gamma}}(z_1, \ldots, z_d) = \sum_{\emptyset \ne \mathfrak{u} \subseteq \{1:d\}} \gamma_{\mathfrak{u}} \Big(\frac{1}{N} \sum_{k=0}^{N-1} \prod_{j \in \mathfrak{u}} B_2\Big(\Big\{\frac{kz_j}{N}\Big\}\Big)\Big),$$

step by step for each $d = 2, 3, \ldots, s$. *In this case, the CBC algorithm is extensible in* s.

(iii) *In the anchored case with general nonproduct weights, the CBC algorithm minimizes an auxiliary quantity depending on* s,

$$\widetilde{e}^2_{N,d,\widetilde{\boldsymbol{\gamma}}_s}(z_1, \ldots, z_d) := \sum_{\emptyset \ne \mathfrak{v} \subseteq \{1:d\}} \widetilde{\gamma}_{s,\mathfrak{v}} \Big(\frac{1}{N} \sum_{k=0}^{N-1} \prod_{j \in \mathfrak{v}} B_2\Big(\Big\{\frac{kz_j}{N}\Big\}\Big)\Big),$$

step by step for each $d = 2, 3, \ldots, s$, *with auxiliary weights defined by*

$$\widetilde{\gamma}_{s,\mathfrak{v}} := \sum_{\mathfrak{v} \subseteq \mathfrak{u} \subseteq \{1:s\}} \gamma_{\mathfrak{u}}\, m^{|\mathfrak{u}| - |\mathfrak{v}|}, \qquad \mathfrak{v} \subseteq \{1:s\}.$$

In this case, the CBC algorithm is not extensible in s.

Outline of the proof. The CBC error bound with product weights is proved in the references [5, 30]. The original proof for Theorem 4.1 remains valid for the unanchored case (with product weights or general nonproduct weights), since in this

https://doi.org/10.1017/S144618111300014X Published online by Cambridge University Press

case the introduction of auxiliary weights is unnecessary. For the anchored case with general nonproduct weights, we proceed as in the original proof to obtain

$$e^2_{N,s,\boldsymbol{\gamma}}(\boldsymbol{z}) = \widetilde{e}^2_{N,s,\widetilde{\boldsymbol{\gamma}}_s}(\boldsymbol{z}).$$

Then we prove by induction that the CBC construction based on the auxiliary quantity yields, for each $d = 1, 2, \ldots, s$,

$$\widetilde{e}^2_{N,d,\widetilde{\boldsymbol{\gamma}}_s}(z_1, \ldots, z_d) \le \bigg(\sum_{\emptyset \ne \mathfrak{v} \subseteq \{1:d\}} \widetilde{\gamma}^{\lambda}_{s,\mathfrak{v}} \bigg(\frac{2\zeta(2\lambda)}{(2\pi^2)^{\lambda}} \bigg)^{|\mathfrak{v}|} \bigg)^{1/\lambda} [\phi(N)]^{-1/\lambda}$$

for all $\lambda \in (1/2, 1]$. Since the auxiliary weights depend only on the maximal dimension s and do not change with the induction index d, we now indeed have the recursive expression

$$\widetilde{e}^2_{N,d,\widetilde{\boldsymbol{\gamma}}_s}(z_1, \ldots, z_{d-1}, z_d) = \widetilde{e}^2_{N,d-1,\widetilde{\boldsymbol{\gamma}}_s}(z_1, \ldots, z_{d-1}) + \theta(z_d),$$

where (suppressing the dependence of θ on $z_1, \ldots, z_{d-1}$)

$$\theta(z_d) := \sum_{d \in \mathfrak{v} \subseteq \{1:d\}} \frac{\widetilde{\gamma}_{s,\mathfrak{v}}}{(2\pi^2)^{|\mathfrak{v}|}} \bigg(\sum_{h_d \in \mathbb{Z}\setminus\{0\}} \frac{1}{h_d^2} \sum_{\substack{\boldsymbol{h}_{\mathfrak{v}\setminus\{d\}} \in (\mathbb{Z}\setminus\{0\})^{|\mathfrak{v}|-1} \\ \boldsymbol{h}_{\mathfrak{v}\setminus\{d\}} \cdot \boldsymbol{z}_{\mathfrak{v}\setminus\{d\}} \equiv -h_d z_d \pmod N}} \frac{1}{\prod_{j \in \mathfrak{v}\setminus\{d\}} h_j^2} \bigg).$$

The proof can then be completed following the argument in the original proof.

Revised Theorem 4.3. *A generating vector $\boldsymbol{z}$ can be constructed by a CBC algorithm such that*

$$R_{N,s,\boldsymbol{\gamma}}(\boldsymbol{z}) \le \frac{2}{N} \sum_{\emptyset \ne \mathfrak{u} \subseteq \{1:s\}} \gamma_{\mathfrak{u}}^{1/2} (c_{\mathrm{lat}} \ln N)^{|\mathfrak{u}|},$$

where

$$c_{\mathrm{lat}} := \sup_{N \ge 2} \bigg\{ \frac{1}{\ln N} + 2 + \frac{2\pi^2(N-1)}{3\,\phi(N) \ln N} \bigg\}.$$

The CBC algorithm minimizes an auxiliary quantity depending on s,

$$\widetilde{R}_{N,d,\widetilde{\boldsymbol{\gamma}}_s}(z_1, \ldots, z_d) := \sum_{\emptyset \ne \mathfrak{v} \subseteq \{1:d\}} \widetilde{\gamma}^{1/2}_{s,\mathfrak{v}} \widetilde{R}_N(\boldsymbol{z}_{\mathfrak{v}}),$$

step by step for each $d = 2, 3, \ldots, s$, with auxiliary weights defined by

$$\widetilde{\gamma}^{1/2}_{s,\mathfrak{v}} := \sum_{\mathfrak{v} \subseteq \mathfrak{u} \subseteq \{1:s\}} \gamma_{\mathfrak{u}}^{1/2}, \qquad \mathfrak{v} \subseteq \{1:s\}.$$

In this case, the CBC algorithm is not extensible in s.

https://doi.org/10.1017/S144618111300014X Published online by Cambridge University Press

Remark

The discussion in Section 5 on the fast CBC construction for product and order dependent (POD) weights is valid for the given generic criterion with POD weights, including the unanchored Sobolev space with POD weights. However, for the anchored Sobolev space with POD weights, the CBC algorithm minimizes an auxiliary quantity that depends on auxiliary weights, but the POD form for the original weights is not preserved in the auxiliary weights. In this sense, Section 5 does not apply to the anchored Sobolev space. Similarly, Section 5 does not apply to the CBC construction based on weighted R. It is therefore our recommendation to work with the unanchored Sobolev space whenever possible.

We now address the computational cost of the fast CBC construction for POD weights. In each step of the construction, there is a "search" cost of $O(N \log N)$ operations which corresponds to the use of fast Fourier transforms for a matrix–vector multiplication, and there is an "update" cost of $O(dN)$ operations at step d which is needed for recursively accumulating a required sum. The construction cost is therefore

$$O\Big(\sum_{d=1}^{s}(N \log N + dN)\Big) = O(sN \log N + s^2 N)$$

operations, with $O(sN)$ storage requirement. If the POD weights are of *finite order* $q \ll s$, then the cost is reduced to $O(sN \log N + qsN)$ operations, with $O(qN)$ storage requirement.

Acknowledgement

We are grateful to Mario Hefter and Steffen Omland from Kaiserslautern for pointing out the error in our proof.

References

[5] J. Dick, "On the convergence rate of the component-by-component construction of good lattice rules", *J. Complexity* **20** (2004) 493–522; doi:10.1016/j.jco.2003.11.008.

[30] F. Y. Kuo, "Component-by-component constructions achieve the optimal rate of convergence in weighted Korobov and Sobolev spaces", *J. Complexity* **19** (2003) 301–320; doi:10.1016/S0885-064X(03)00006-2.