How Many Random Bits Do We Need for Monte Carlo Integration?

Stefan Heinrich[1], Erich Novak[2], and Harald Pfeiffer[2]

[1] Universität Kaiserslautern, FB Informatik, Postfach 3049, D-67653 Kaiserslautern, Germany, heinrich@informatik.uni-kl.de
[2] Universität Jena, Mathematisches Institut, Ernst-Abbe-Platz 2, D-07740 Jena, Germany, novak@minet.uni-jena.de, pfeiffha@minet.uni-jena.de

Summary. We study Monte Carlo methods (randomized algorithms) that use only a small number of random bits instead of more general random numbers for the computation of sums and integrals. To approximate $N^{-1}\sum_{i=0}^{N-1} f_i$ for $f \in \mathbf{R}^N$, the classical Monte Carlo method uses n function values, that is, coordinates of f, and n random numbers. Our method gives the same error with only $2\lceil \log_2 N \rceil$ random bits, independently of n. To approximate $\int_{[0,1]^d} f(x)\,dx$ for f from a Sobolev space, the classical Monte Carlo method uses n function values and $d \cdot n$ random numbers. We present a method with the optimal order of convergence that uses only at most $(2+d)\log_2 n$ random bits.

1 Introduction

There is a long tradition to study "optimal" Monte Carlo methods (randomized algorithms). One first should mention the work of Bakhvalov ([2], [3], [4]). A typical result is the following. Let f be from the Sobolev space $W_p^r([0,1]^d)$ with $r \cdot p > d$. If n function evaluations are allowed to compute

$$I_d(f) = \int_{[0,1]^d} f(x)\,dx,$$

the optimal rate of convergence of deterministic methods is $n^{-r/d}$ while that of randomized algorithms is $n^{-r/d-1/2}$ for $p \geq 2$ and $n^{-r/d-1+1/p}$ for $1 \leq p < 2$. Hence optimal Monte Carlo methods are much better for $p > 1$ and large d. Optimality refers to the relation between the number of function evaluations and the error of a method.

As a source of randomness a method typically uses random numbers from $[0,1]$, and most of the classical methods use about $d \cdot n$ of those random numbers. Bakhvalov ([4], [5]), again, observed that such a huge amount of randomness is not really necessary. It suffices to use pairwise independent sample points for the Monte Carlo integration of a function $f : [0,1]^d \to \mathbf{R}$.

With $n^{-1}\sum_{k=1}^{n} f(x_k)$, where $x_k = x + k\,y \mod 1$, and x and y are independent, uniformly distributed on $[0,1]^d$ random variables, one obtains the same error as with the classical Monte Carlo method. The latter uses $d \cdot n$, the former, however, only $2\,d$ random numbers (for the generation of x and y).

In this paper we replace random numbers from $[0,1]$ by random bits from $\{0,1\}$ and we want to construct randomized algorithms that use only a few random bits for the computation of sums and integrals. We will prove that a rather small amount of randomness suffices to obtain the optimal order of convergence. To approximate a sum $N^{-1}\sum_{i=0}^{N-1} f_i$ for $f \in \mathbf{R}^N$, the classical Monte Carlo method uses n function values, i. e., coordinates of f, and n random numbers from $\{0,\dots,N-1\}$. Our method gives the same error with only $2\lceil \log_2 N \rceil$ random bits, independently of n or the accuracy of the method. This result is applied to the computation of integrals. To compute $\int_{[0,1]^d} f(x)\,dx$ for f from a Sobolev space, the classical Monte Carlo method uses n function values and $d \cdot n$ random numbers. We present a method with the optimal order of convergence that uses at most $(2+d)\log_2 n$ random bits.

Hence one needs only about $d \log n$ random bits to obtain the optimal order of convergence for the integration of functions in Sobolev spaces. The analogous result also holds for Hölder spaces. So, by the techniques of the present paper, one can improve the respective results of Novak [21].

We end this introduction with a few historical remarks. As mentioned, randomized algorithms in numerical analysis and continuous mathematics tend to use random numbers from $[0,1]$, or an even more general source of randomness. See Heinrich [11], Novak [18], and Traub, Wasilkowski, Woźniakowski [23] for results on the complexity of numerical problems in the randomized setting. In computer science and in discrete mathematics one tends to use random bits as a source of randomness, see Motwani, Raghavan [16].

There are relatively few papers and books that discuss the use of random bits for continuous problems and compare them with a more general randomness. See Blum, Cucker, Shub, Smale [7], Novak ([17], [18], [19], [21]) and Traub, Woźniakowski [24]. The use of random bits for the summation problem and for related problems was studied in Chor, Goldreich [8], Goldreich, Wigderson [9], and Joffe [13].

2 The Summation Problem

We are interested in the approximate computation of a mapping

$$S : F \to \mathbf{R}, \tag{1}$$

where F is a class of real-valued functions on a set D and S is an arbitrary mapping – the "solution operator", mapping an input (instance) $f \in F$ of our numerical problem to the exact solution $S(f)$. In this paper we consider S being either the operator of taking the mean of a finite sequence, or the

integral over the d-dimensional unit cube. In the present section, we deal with the former, the latter is considered in Section 3.

More precisely, we consider the following problem. Compute, for $f \in \mathbf{R}^N$ (so here we have $D = \{0, \ldots, N-1\}$),

$$S_N(f) = \frac{1}{N} \sum_{i=0}^{N-1} f_i \,.$$

We assume that f is from the set $F = \mathcal{B}(L_p^N)$ of all $f \in \mathbf{R}^N$ such that

$$\frac{1}{N} \sum_{i=0}^{N-1} |f_i|^p \le 1$$

for $1 \le p < \infty$ and

$$|f_i| \le 1, \qquad i \in \{0, 1, \ldots, N-1\}$$

for $p = \infty$. We use the norm

$$\|f\|_p = \left(\frac{1}{N} \sum_{i=0}^{N-1} |f_i|^p \right)^{1/p},$$

with the usual modification for $p = \infty$. To write $f(i)$ instead of f_i is sometimes more convenient; we use both notations.

We study deterministic and randomized algorithms for the computation of $S_N(f)$ up to some error $\varepsilon > 0$. We use the real number model of computation, with unit cost for each arithmetic operation, with an "oracle" (or "subroutine") which, at our request, for $i \in D$ supplies the value $f(i)$. In addition we allow for general Monte Carlo methods the instruction "choose a random number from $[0,1]$"and for restricted Monte Carlo algorithms the instruction "flip a coin" or "choose randomly an element from $\{0,1\}$", and also the cost of these instructions is one. See Novak [19] for details about the model of computation. We thus obtain three different cases as far as the "allowed" randomness of the algorithms is concerned. In this paper we are mainly interested in the case of *restricted Monte Carlo methods.*

2.1 Deterministic Algorithms

We start with deterministic algorithms for the approximation of a given solution operator S as in (1). They are of the form

$$A_n(f) = \varphi(f(i_1), \ldots, f(i_n)) \,, \tag{2}$$

where $i_k \in D$ for $k = 1, \ldots, n$ and φ is a real-valued mapping on $\mathbf{R}^n$. In the terminology of information-based complexity, these algorithms constitute

the class of all nonadaptive, in general nonlinear algorithms using n function values. Linear algorithms

$$A_n^{\rm lin}(f) = \sum_{k=1}^{n} a_k\, f(i_k) \tag{3}$$

with $a_k \in \mathbf{R}$ and $i_k \in D$ for $k = 1, \ldots, n$ form a special subclass.

The error of a method A_n of the form (2) on F is defined as

$$e(S, A_n, F) = \sup_{f \in F} |S(f) - A_n(f)|\,.$$

The central quantity for our analysis is the nth minimal error given by

$$e_n^{\rm det}(S, F) = \inf_{A_n} e(S, A_n, F)\,.$$

Formally one can also allow the case $n = 0$ with constant algorithms A_0. Since we have $e_0^{\rm det}(S, F) = 1$ for the problems we study here, they all are scaled in the same way. Of course we obtain $e_n^{\rm det}\big(S_N, \mathcal{B}(L_p^N)\big) = 0$ for $n \geq N$. We therefore always assume that $n < N$ when considering $S = S_N$.

The classes $\mathcal{B}(L_p^N)$ are unit balls in Banach spaces, so they are convex and symmetric. The functional S_N is linear. Under these assumptions linear methods of the form (3) are known to be optimal (even among all adaptive, nonlinear methods). This result of Smolyak and Bakhvalov is proved in Bakhvalov [6], see also Novak [20], and Traub, Wasilkowski, Woźniakowski [23]. Consequently, one easily finds an optimal method for the summation operator S_N on $\mathcal{B}(L_p^N)$, for instance,

$$A_n^*(f) = \frac{1}{N} \sum_{i=0}^{n-1} f_i \tag{4}$$

with error

$$e\big(S_N, A_n^*, \mathcal{B}(L_p^N)\big) = \left(\frac{N-n}{N}\right)^{1-1/p} \qquad (n < N).$$

The spaces $\mathcal{B}(L_p^N)$ are increasing with decreasing p and for the extreme cases $p = \infty$ and $p = 1$ we obtain

$$e_n^{\rm det}\big(S_N, \mathcal{B}(L_\infty^N)\big) = \frac{N-n}{N} \qquad \text{and} \qquad e_n^{\rm det}\big(S_N, \mathcal{B}(L_1^N)\big) = 1 \qquad (n < N)\,.$$

For later reference we summarize these (well known) results.

Theorem 1. *Let $1 \leq p \leq \infty$ and $n < N$. Then*

$$e_n^{\rm det}\big(S_N, \mathcal{B}(L_p^N)\big) = \left(\frac{N-n}{N}\right)^{1-1/p}.$$

What does this mean for the complexity of the summation problem? Let us recall the related notion in the general setting (1). We assume that the cost of one function evaluation (oracle call) is $\mathbf{c}$ and that of one arithmetic operation is one. The cost of a deterministic algorithm is the weighted sum of the number of performed arithmetic operations (with weight one) and the number of function evaluations (with weight $\mathbf{c}$). For $\varepsilon > 0$, the ε-complexity $\mathrm{comp}_\varepsilon^{\mathrm{det}}(S, F)$ is the minimal cost of a deterministic algorithm A with $e\big(S, A, F\big) \le \varepsilon$.

The next result follows easily from Theorem 1, together with the special form (4) of the optimal A_n^*.

Theorem 2. *Let* $1 \le p \le \infty$ *and* $0 < \varepsilon < 1$. *Then*

$$\mathbf{c}\,\lceil N(1-\varepsilon^{p/(p-1)})\rceil \le \mathrm{comp}_\varepsilon^{\mathrm{det}}(S_N, \mathcal{B}(L_p^N)) \le (\mathbf{c}+1)\,\lceil N(1-\varepsilon^{p/(p-1)})\rceil\,. \quad (5)$$

Here we put $\varepsilon^{p/(p-1)} = 0$ *for* $p = 1$ *and* $\varepsilon^{p/(p-1)} = \varepsilon$ *for* $p = \infty$.

Observe that in all cases the complexity is proportional to N, the number of summands.

2.2 General Randomized Algorithms

For a (general) randomized algorithm for the approximation of S as in (1) we allow in addition to the operations of the real number model of computation with an oracle the instruction "choose a number from $[0,1]$ according to the Lebesgue measure".

Let $\big([0,1], \mathcal{B}([0,1]), \lambda_{|[0,1]}\big)$ be the probability space that corresponds to this instruction, that is, $\mathcal{B}([0,1])$ is the Borel σ-algebra on $[0,1]$ and $\lambda_{|[0,1]}$ the restriction of the Lebesgue measure on $[0,1]$, and let $(\Omega, \mathcal{B}, \mathbf{P})$ be the countable infinite product of $\big([0,1], \mathcal{B}([0,1]), \lambda_{|[0,1]}\big)$.

To every randomized algorithm A we associate for each $\omega = (\omega_1, \omega_2, \ldots) \in \Omega$ a (partial) mapping $A_\omega : F \to \mathbf{R}$ as follows. Given a problem element $f \in F$, we apply A to f taking, when necessary, ω_i as the ith random number. If the algorithm terminates, we set $A_\omega(f)$ equal to the output of the algorithm. Furthermore, let $e(S, A, f, \omega) = |S(f) - A_\omega(f)|$, and let $\mathrm{cost}(A, f, \omega)$ be the minimal value of the weighted sum of the number of performed arithmetic operations (with weight one), of the number of function evaluations (with weight $\mathbf{c}$), and of the number of used entries of ω (with weight one).

We shall concentrate on randomized algorithms A such that the mappings $e(S, A, f, .)$, $\mathrm{cost}(A, f, .) : \Omega \to \mathbf{R}_0^+$ are defined almost everywhere and measurable for each $f \in F$. This is not really a limitation, since each "reasonable" algorithm fulfills these conditions. We call the well defined quantities

$$e(S, A, f) := \Big(\mathbf{E}\big(e(S, A, f, .)^2\big)\Big)^{1/2} \quad (6)$$

and

$$\mathrm{cost}(A,f) := \mathbf{E}\big(\mathrm{cost}(A,f,.)\big)$$

the individual error of A for f and the individual cost of A for f, respectively. Here, $\mathbf{E}$ denotes the expectation with respect to $\mathbf{P}$. Furthermore, we call

$$e(S,A,F) := \sup_{f\in F} e(S,A,f)$$

and

$$\mathrm{cost}(A,F) := \sup_{f\in F} \mathrm{cost}(A,f)$$

the error of A on F and the cost of A on F, respectively. The ε-complexity of the problem S on F in the randomized setting is given by

$$\mathrm{comp}_\varepsilon^{\mathrm{ran}}(S,F) := \inf\big\{\mathrm{cost}\big(A,F\big) :\ A \text{ randomized algorithm},\ e\big(S,A,F\big) \le \varepsilon\big\}.$$

We are especially interested in randomized algorithms $A_n = (A_{n,\omega})_{\omega\in\Omega}$,

$$A_{n,\omega}(f) = \varphi^\omega(f(i_1^\omega),\dots,f(i_n^\omega))\,,$$

that use n function values. Here, $i_1^\omega,\dots,i_n^\omega$ are elements of D and φ^ω maps $\mathbf{R}^n$ to $\mathbf{R}$ such that for each $f\in F$ the mapping $\omega\mapsto A_{n,\omega}(f)$ is measurable. Again, randomized linear algorithms are a special case,

$$A_{n,\omega}^{\mathrm{lin}}(f) = \sum_{k=1}^n a_k^\omega\, f(i_k^\omega)\,. \tag{7}$$

The randomized nth minimal error is defined as

$$e_n^{\mathrm{ran}}(S,F) = \inf_{A_n} e(S,A_n,F)\,.$$

Let us mention that in contrast to the deterministic setting no general result about the optimality of linear methods among all methods holds for the randomized setting. Mathé [15] found the optimal randomized summation algorithm for $2\le p\le\infty$. It does not depend on p and has the following form:

- Choose an n-subset $\{i_1^\omega,\dots,i_n^\omega\}\subset\{0,1,\dots,N-1\}$ according to the equidistribution on the family of all $\binom{N}{n}$ n-subsets.
- Put

$$\mathcal{A}_{n,\omega}^2(f) = c\sum_{k=1}^n f(i_k^\omega)\,,$$

where

$$c = \left(n+\sqrt{\frac{n(N-n)}{N-1}}\right)^{-1}. \tag{8}$$

We call this the "algorithm $\mathcal{A}_n^2$ with constant c", also for other positive values of c. The c from (8) satisfies

$$\frac{1}{n+\sqrt{n}} \le c \le \frac{1}{n}\,.$$

Note also that for $2 \le p \le \infty$ the optimal method is linear. We summarize the known results on $e_n^{\rm ran}\big(S_N, \mathcal{B}(L_p^N)\big)$, see Mathé [15].

Below we use asymptotic notation $a(n,N) \prec b(n,N)$ for non-negative real functions $a(n,N)$ and $b(n,N)$, which means that there exist some constant $c > 0$ and some $n_0, N_0 \in \mathbf{N}$ such that $a(n,N) \le c\, b(n,N)$ for all $n \ge n_0$, $N \ge N_0$. If $a(n,N) \prec b(n,N)$ and $b(n,N) \prec a(n,N)$, then we write $a(n,N) \asymp b(n,N)$. We use notation with analogous meaning also for functions of other variables, which should be clear from the context. Also, we often use the same symbol c for possibly different constants.

Theorem 3. *Let $2 \le p \le \infty$. Then, for $n < N$,*

$$e_n^{\rm ran}\big(S_N, \mathcal{B}(L_p^N)\big) = \frac{1}{1+\sqrt{\frac{(N-1)n}{N-n}}}\,. \tag{9}$$

For $1 \le p < 2$ only the order of the error is known: Let $\beta > 1$. Then, for $\beta n < N$,

$$e_n^{\rm ran}\big(S_N, \mathcal{B}(L_p^N)\big) \asymp n^{-1+1/p}\,. \tag{10}$$

In particular it follows from (9) that

$$e_n^{\rm ran}\big(S_N, \mathcal{B}(L_p^N)\big) \asymp n^{-1/2}$$

for $2 \le p \le \infty$ and $\beta n < N$, where $\beta > 1$.

We want to mention a couple of interesting facts around relation (10). One might ask whether the classical Monte Carlo method, we call it $\mathcal{A}_n^1$ for short, yields this optimal rate for $1 < p < 2$. It does not. The reason is that the variances of functions in $\mathcal{B}(L_p^N)$, which we need for the error as defined in (6), will not be bounded uniformly in N. Even more is true: no linear method (7) can reach this rate. Mathé [14] proved that for $1 \le p < 2$ and $\beta n < N$ the error of optimal linear methods is of the order

$$\min(N^{1/p-1/2} n^{-1/2}, 1)\,.$$

However, it is easily checked that a slight (nonlinear) modification of the classical Monte Carlo method does give the optimal rate: replace the vector $f \in \mathcal{B}(L_p^N)$ by $\tilde{f}$ defined by $\tilde{f}_i = f_i$ if $|f_i| \le n^{1/p}$ and $\tilde{f}_i = 0$ otherwise. Then apply the classical Monte Carlo method $\mathcal{A}_n^1$ to $\tilde{f}$.

The lower bounds (in (10) and in Theorem 4) are well known. They can be proved by applying the results of Sect. 2.2.4 from Novak [18]. Again we ask: what do we get for the complexity of the summation problem? The cost of the algorithm $\mathcal{A}_n^1$ is proportional to n and the same is true for its simple nonlinear modification. We therefore obtain the following result.

Theorem 4. *Let $1 \le p \le \infty$. Then, for $0 < \varepsilon < 1$ and $N \in \mathbf{N}$,*

$$\mathrm{comp}_\varepsilon^{\mathrm{ran}}(S_N, \mathcal{B}(L_p^N)) \asymp \mathbf{c} \cdot \min(N, \varepsilon^{-2}), \quad \textit{if} \quad p \ge 2,$$

and

$$\mathrm{comp}_\varepsilon^{\mathrm{ran}}(S_N, \mathcal{B}(L_p^N)) \asymp \mathbf{c} \cdot \min(N, \varepsilon^{-p/(p-1)}), \quad \textit{if} \quad 1 < p < 2.$$

For $p = 1$ we obtain

$$\mathrm{comp}_\varepsilon^{\mathrm{ran}}(S_N, \mathcal{B}(L_1^N)) \asymp \mathbf{c}\, N\, (1 - \varepsilon^2).$$

2.3 Restricted Randomized Algorithms

For restricted randomized algorithms we allow in addition to the operations of the real number model of computation with an oracle only the instruction "choose a random bit", i.e., "choose an element of $\{0,1\}$ according to the equidistribution", instead of the more general instruction "choose a number from $[0,1]$ according to the Lebesgue measure".

We now denote by $(\Omega, \mathcal{B}, \mathbf{P})$ the countable infinite product of the probability space that corresponds to the drawing of a random bit and proceed as in the beginning of Sect. 2.2. The ε-complexity now is given by

$$\mathrm{comp}_\varepsilon^{\mathrm{coin}}(S,F) := \inf\big\{\mathrm{cost}\big(A,F\big) : \ A \text{ restricted randomized algorithm}, \ e\big(S,A,F\big) \le \varepsilon\big\}.$$

In the following we study these numbers for $S = S_N$ and $F = \mathcal{B}(L_p^N)$ and start with algorithms to obtain good upper bounds. Of course, we can implement the algorithms $\mathcal{A}_n^1$ and $\mathcal{A}_n^2$ from Sect. 2.2 and also their nonlinear modifications, which we use for $1 < p < 2$, as restricted randomized algorithms. What is the cost of these algorithms? For $\mathcal{A}_n^1$ we have to sample uniformly from the set $\{0, 1, \ldots, N-1\}$. We can achieve this by the following procedure.

Let $s \in \mathbf{N}$ be such that $2^{s-1} < N \le 2^s$. (Of course, we assume $N > 1$.)

- Step 1: Choose independently s random bits and view them (in a fixed order) as binary representation of a number, z, say.
- Step 2: If $z \in \{0, 1, \ldots, N-1\}$, stop and take z as output, otherwise go back to step 1.

For fixed $x \in \{0, 1, \ldots, N-1\}$ the probability of getting x as output is obviously

$$\frac{1}{2^s} + \frac{2^s - N}{2^s}\frac{1}{2^s} + \left(\frac{2^s - N}{2^s}\right)^2 \frac{1}{2^s} + \cdots = \frac{1}{2^s} \sum_{i=0}^{\infty} \left(\frac{2^s - N}{2^s}\right)^i = \frac{1}{N}.$$

Hence the above procedure implements indeed uniform sampling from the set $\{0, 1, \ldots, N-1\}$. How many random bits on the average are necessary to

realize an element of $\{0,1,\dots,N-1\}$ according to the uniform distribution? Clearly, this number equals

$$s\frac{N}{2^s}+2s\frac{2^s-N}{2^s}\frac{N}{2^s}+3s\left(\frac{2^s-N}{2^s}\right)^2\frac{N}{2^s}+\cdots=s\frac{N}{2^s}\sum_{i=1}^{\infty}i\left(\frac{2^s-N}{2^s}\right)^{i-1}=s\frac{2^s}{N},$$

it is of the order $\log N$. These considerations show that we can implement algorithm $\mathcal{A}_n^1$ as a restricted randomized algorithm with cost

$$\mathrm{cost}(\mathcal{A}_n^1,\mathcal{B}(L_p^N))\asymp \mathbf{c}n+n\log N.$$

The implementation of $\mathcal{A}_n^2$ is more difficult but the final cost is of the same order, at least in the interesting case where N is much bigger than n. Instead of the same upper bounds as in Theorem 4 we now obtain only

$$\mathrm{comp}_\varepsilon^{\mathrm{coin}}(S_N,\mathcal{B}(L_p^N))\prec \min(\mathbf{c}N,\,\varepsilon^{-2}(\mathbf{c}+\log N))\,,\quad \text{if}\quad p\geq 2, \tag{11}$$

and

$$\mathrm{comp}_\varepsilon^{\mathrm{coin}}(S_N,\mathcal{B}(L_p^N))\prec \min(\mathbf{c}N,\,\varepsilon^{-p/(p-1)}(\mathbf{c}+\log N))\,,\quad \text{if}\quad 1<p<2\,. \tag{12}$$

For $p=1$ and a fixed $0<\varepsilon<1$ we obtain again

$$\mathrm{comp}_\varepsilon^{\mathrm{coin}}(S_N,\mathcal{B}(L_1^N))\asymp \mathbf{c}\,N\,. \tag{13}$$

We know from Sect. 2.2 that the algorithms $\mathcal{A}_n^1$ and $\mathcal{A}_n^2$ are (almost) optimal with respect to the number of used function values. If we allow only random bits then the cost of these algorithms is much higher than n. For example, if $\log N$ is of the order n then the cost of these algorithms is of the order n^2. We will see that there are much better algorithms. The additional factor $\log N$ in (11) and (12) is not needed, since there are algorithms with the same error using only about $\log N$ random bits, independently of ε.

For the case that N is a prime number we define two more (linear) algorithms $\mathcal{A}_n^3$ and $\mathcal{A}_n^4$. Of course we can define their nonlinear modifications (for $1<p<2$) as before: replace the vector $f\in\mathcal{B}(L_p^N)$ by $\tilde{f}$ defined by $\tilde{f}_i=f_i$ if $|f_i|\leq n^{1/p}$ and $\tilde{f}_i=0$ otherwise. Then apply $\mathcal{A}_n^i$ to $\tilde{f}$.

We repeat the definitions of $\mathcal{A}_n^1$ and $\mathcal{A}_n^2$ and define $\mathcal{A}_n^3$ and $\mathcal{A}_n^4$ for the case where N is a prime number. All the $\mathcal{A}_n^i$ are of the form

$$\mathcal{A}_{n,\omega}^i(f)=c\sum_{k=1}^{n}f(i_k^\omega)$$

with some $c>0$. Only the choice of the i_k^ω is different.

Algorithm $\mathcal{A}_n^1$: We take the i_k^ω independently according to the uniform distribution on $\{0,1,\dots,N-1\}$.

Algorithm $\mathcal{A}_n^2$: We choose randomly an n-subset $\{i_1^\omega, \dots, i_n^\omega\} \subset \{0, 1, \dots, N-1\}$, equidistributed on the family of all $\binom{N}{n}$ n-subsets.

Algorithm $\mathcal{A}_n^3$: We take $x^\omega, y^\omega \in \{0, 1, \dots, N-1\}$ independently according to the uniform distribution and put

$$i_k^\omega = x^\omega + (k-1) \cdot y^\omega \mod N .$$

Algorithm $\mathcal{A}_n^4$: We choose $x^\omega \in \{0, 1, \dots, N-1\}$ and $y^\omega \in \{1, 2, \dots, N-1\}$ independently according to the respective uniform distribution and put

$$i_k^\omega = x^\omega + (k-1) \cdot y^\omega \mod N .$$

From our considerations above we know that we can implement the algorithms $\mathcal{A}_n^3$ and $\mathcal{A}_n^4$ using (on the average) about $\log N$ random bits. Hence we obtain

$$\mathrm{cost}(\mathcal{A}_n^3, \mathcal{B}(L_p^N)) \asymp \mathrm{cost}(\mathcal{A}_n^4, \mathcal{B}(L_p^N)) \asymp \mathbf{c}n + \log N .$$

We now turn to the individual error of these four algorithms.

Lemma 1. *Let $f \in F$. For $\mathcal{A}_n^1$ and $\mathcal{A}_n^3$ we obtain*

$$e(S_N, \mathcal{A}_n^1, f)^2 = e(S_N, \mathcal{A}_n^3, f)^2 = \sum_{i=0}^{N-1} f_i^2 \left(\frac{1 - 2cn + c^2 n(n-1)}{N^2} + \frac{c^2 n}{N} \right) + \frac{1}{N^2} \sum_{i \neq j} f_i f_j \big(1 - 2cn + c^2 n(n-1)\big) . \quad (14)$$

In particular, if $c = 1/n$, we have

$$e(S_N, \mathcal{A}_n^1, f)^2 = e(S_N, \mathcal{A}_n^3, f)^2 = \frac{1}{\sqrt{n}} \frac{1}{N} \sum_{i=0}^{N-1} \big(S_N(f) - f_i\big)^2 . \quad (15)$$

For $\mathcal{A}_n^2$ and $\mathcal{A}_n^4$ we obtain

$$e(S_N, \mathcal{A}_n^2, f)^2 = e(S_N, \mathcal{A}_n^4, f)^2 = \sum_{i=0}^{N-1} f_i^2 \left(\frac{1 - 2cn}{N^2} + \frac{c^2 n}{N} \right) + \sum_{i \neq j} f_i f_j \left(\frac{1 - 2cn}{N^2} + \frac{c^2 n(n-1)}{N(N-1)} \right) . \quad (16)$$

In particular, if $c = \big(n + \sqrt{n(N-n)/(N-1)}\big)^{-1}$, we have

$$e(S_N, \mathcal{A}_n^2, f)^2 = e(S_N, \mathcal{A}_n^4, f)^2 = \left(1 + \sqrt{\frac{n(N-1)}{N-n}} \right)^{-2} \frac{1}{N} \sum_{i=0}^{N-1} f_i^2 . \quad (17)$$

Proof. A straightforward computation under use of standard properties of (pairwise) independent random variables gives (14) for algorithm $\mathcal{A}_n^1$. The same computation also yields (14) for $\mathcal{A}_n^3$, since we assume that N is a prime number and we therefore have the representation $\mathcal{A}_n^3(f) = c\sum_{k=1}^n f(X_k)$, where $X_1, \ldots, X_n$ are pairwise independent and on the set $\{0, 1, \ldots, N-1\}$ equidistributed random variables, see Remark 1.

If $c = 1/n$, algorithm $\mathcal{A}_n^1$ is just the classical Monte Carlo method and (15) is its well known error formula. Again, the same reasoning that leads to this formula is also valid for algorithm $\mathcal{A}_n^3$.

We turn to algorithm $\mathcal{A}_n^2$. Let P_n be the set of all n-subsets of $\{0, 1, \ldots, N-1\}$. Using the notation χ_I for the indicator function of a set I, we have

$$\begin{aligned} e(S_N, \mathcal{A}_n^2, f)^2 &= \frac{1}{\binom{N}{n}} \sum_{I \in P_n} \left(\sum_{i=0}^{N-1} f_i \Big(\frac{1}{N} - c\,\chi_I(i) \Big) \right)^2 \\ &= \frac{1}{\binom{N}{n}} \sum_{i=0}^{N-1} f_i^2 \sum_{I \in P_n} \Big(\frac{1}{N} - c\,\chi_I(i) \Big)^2 \\ &\quad + \frac{1}{\binom{N}{n}} \sum_{i \neq j} f_i f_j \sum_{I \in P_n} \Big(\frac{1}{N} - c\,\chi_I(i) \Big) \Big(\frac{1}{N} - c\,\chi_I(j) \Big) . \end{aligned}$$

Fix $i, j \in \{0, 1, \ldots, N-1\}$, $i \neq j$. Then there are $\binom{N-1}{n-1}$ elements of P_n that contain i and j, respectively, and $\binom{N-2}{n-2}$ elements that contain both i and j. It follows that

$$\begin{aligned} e(S_N, \mathcal{A}_n^2, f)^2 &= \frac{1}{\binom{N}{n}} \sum_{i=0}^{N-1} f_i^2 \left(\binom{N}{n} \frac{1}{N^2} - \frac{2c}{N} \binom{N-1}{n-1} + c^2 \binom{N-1}{n-1} \right) \\ &\quad + \frac{1}{\binom{N}{n}} \sum_{i \neq j} f_i f_j \left(\binom{N}{n} \frac{1}{N^2} - \frac{2c}{N} \binom{N-1}{n-1} + c^2 \binom{N-2}{n-2} \right) . \end{aligned}$$

This implies (16) for $\mathcal{A}_n^2$.

Now we consider algorithm $\mathcal{A}_n^4$ supposing that N is a prime number. We define, for $x = 0, 1, \ldots, N-1$ and $y = 1, 2, \ldots, N-1$, the set

$$I_n^{x,y} := \{x + (k-1) \cdot y \mod N : k = 1, \ldots, n\}.$$

Since N is prime, the elements of $I_n^{x,y}$ are pairwise distinct. We obtain

$$\begin{aligned} e(S_N, \mathcal{A}_n^4, f)^2 &= \frac{1}{N(N-1)} \sum_{\substack{0 \le x \le N-1 \\ 1 \le y \le N-1}} \left(\sum_{i=0}^{N-1} f_i \Big(\frac{1}{N} - c\,\chi_{I_n^{x,y}}(i) \Big) \right)^2 \\ &= \frac{1}{N(N-1)} \left(\sum_{i=0}^{N-1} f_i^2 \sum_{\substack{0 \le x \le N-1 \\ 1 \le y \le N-1}} \Big(\frac{1}{N} - c\,\chi_{I_n^{x,y}}(i) \Big)^2 \right. \end{aligned}$$

$$+\sum_{i\neq j} f_i f_j \sum_{\substack{0\le x\le N-1\\ 1\le y\le N-1}} \Big(\frac{1}{N} - c\,\chi_{I_n^{x,y}}(i)\Big)\Big(\frac{1}{N} - c\,\chi_{I_n^{x,y}}(j)\Big)\Big).$$

Fix $i,j \in \{0,1,\dots,N-1\}$, $i \neq j$. For $k = 1,\dots,n$ there exists for each $y \in \{1,2,\dots,N-1\}$ exactly one $x \in \{0,1,\dots,N-1\}$ such that $i = x + (k-1)\cdot y \mod N$. Hence the number of sets $I_n^{x,y}$ that contain i equals $n(N-1)$, and the same holds for j. How many of the sets $I_n^{x,y}$ contain both i and j? Observe that, since N is prime, for each pair $k,l \in \{1,\dots,n\}$ with $k \neq l$ there is exactly one pair x,y with

$$x + (k-1)y = i \mod N, \quad x + (l-1)y = j \mod N.$$

So there are exactly $n(n-1)$ sets $I_n^{x,y}$ that contain both i and j. It follows that

$$e(S_N,\mathcal{A}_n^4,f)^2 = \frac{1}{N(N-1)}\Bigg(\sum_{i=0}^{N-1} f_i^2\Big(\frac{N(N-1)}{N^2} - \frac{2cn(N-1)}{N} + n(N-1)c^2\Big) + \sum_{i\neq j} f_i f_j\Big(\frac{N(N-1)}{N^2} - \frac{2cn(N-1)}{N} + n(n-1)c^2\Big)\Bigg).$$

This gives (16) for algorithm $\mathcal{A}_n^4$.

Finally, if $c = \big(n + \sqrt{n(N-n)/(N-1)}\big)^{-1}$, it is easy to check that

$$\frac{1-2cn}{N^2} + \frac{c^2 n}{N} = \left(1+\sqrt{\frac{n(N-1)}{N-n}}\right)^{-2}\frac{1}{N} \quad \text{and} \quad \frac{1-2cn}{N^2} + \frac{c^2 n(n-1)}{N(N-1)} = 0\,.$$

This, together with (16), shows (17). □

Remark 1. a) The error of all four methods is similar if n is much smaller than N. The cost of $\mathcal{A}_n^1$ and $\mathcal{A}_n^2$ is about $n\cdot(\mathbf{c}+\log N)$, the cost of $\mathcal{A}_n^3$ and $\mathcal{A}_n^4$ is only about $\mathbf{c}n + \log N$.

b) Concerning $\mathcal{A}_n^1$ and $\mathcal{A}_n^3$ we mention that the classical Monte Carlo method is based on the fact that we use pairwise independent indices, a "complete" independence is not needed. It is not difficult to prove the following. Let $N \in \mathbf{N}$, and let X_1 and X_2 be independent and on the set $\{0,1,\dots,N-1\}$ equidistributed random variables. Define, for $k \in \mathbf{N}$,

$$Z_k = X_1 + k\cdot X_2 \mod N.$$

Then Z_k is equidistributed on $\{0,1,\dots,N-1\}$. Furthermore, Z_k and Z_l are independent if and only if

$$\gcd(N,k-l) = 1.$$

c) The idea behind algorithm $\mathcal{A}_n^3$, to construct pairwise independent indices, was independently found by different authors. See Bakhvalov [4], Chor,

Goldreich [8], Goldreich, Wigderson [9], Joffe [13], Sugita, Takanobu [22], and papers mentioned by these authors for more information. Algorithm $\mathcal{A}_n^4$ combines the optimality of algorithm $\mathcal{A}_n^2$ (optimality in the sense error versus number of function values) with the small amount of randomness of $\mathcal{A}_n^3$. This latter algorithm seems to be new.

d) So far we assumed that N is a prime number and we used of the order $\log N$ random bits. For general N we have to modify the algorithms slightly. This is what we do next. Finally, we will consider an algorithm that uses *exactly* $2\lceil \log_2 N \rceil$ random bits.

In the following we always assume $c = \big(n + \sqrt{n(N-n)/(N-1)}\big)^{-1}$ for algorithm $\mathcal{A}_n^4$. Note that the proof of (16) for $\mathcal{A}_n^4$ depends heavily on the fact that N is a prime number. By using $\mathcal{A}_n^4$ we define an algorithm $\widetilde{A}_n^4$ for general N. According to Bertrand's Postulate, see Hardy, Wright [10], p. 343, there is a prime number P such that $N < P \le 2N$. For $f \in \mathbf{R}^N$ we define $f^* \in \mathbf{R}^P$ by

$$f^* := \big(f_0, \dots, f_{N-1}, 0, \dots, 0\big)$$

so that $f \in \mathcal{B}(L_p^N)$ implies $f^* \in \mathcal{B}(L_p^P)$. Then we apply algorithm $\mathcal{A}_n^4$ to f^*. Briefly, applying $\widetilde{A}_n^4$ to f means applying $\mathcal{A}_n^4$ to f^* and multiplying the result with P/N.

It follows from (17) that

$$e\big(S_N, \widetilde{A}_n^4, f\big)^2 = \frac{P^2}{N^2} e\big(S_P, \mathcal{A}_n^4, f^*\big)^2 = \frac{P^2}{N^2}\left(1 + \sqrt{\frac{n(P-1)}{P-n}}\right)^{-2} \frac{1}{P} \sum_{i=0}^{P-1} \big(f_i^*\big)^2$$

$$= \frac{P}{N}\left(1 + \sqrt{\frac{n(P-1)}{P-n}}\right)^{-2} \frac{1}{N} \sum_{i=0}^{N-1} f_i^2.$$

Hence we have

$$e\big(S_N, \widetilde{A}_n^4, \mathcal{B}(L_2^N)\big) = \left(1 + \sqrt{\frac{n(P-1)}{P-n}}\right)^{-1} \sqrt{\frac{P}{N}} \le \sqrt{2}\, n^{-1/2}.$$

We summarize our upper bounds as follows.

Theorem 5. *Let $1 \le p \le \infty$. Then, for $0 < \varepsilon < 1$ and $N \in \mathbf{N}$,*

$$\mathrm{comp}_\varepsilon^{\mathrm{coin}}(S_N, \mathcal{B}(L_p^N)) \prec \min(\mathbf{c}N, \mathbf{c}\varepsilon^{-2} + \log N), \quad \textit{if} \quad p \ge 2,$$

and

$$\mathrm{comp}_\varepsilon^{\mathrm{coin}}(S_N, \mathcal{B}(L_p^N)) \prec \min(\mathbf{c}N, \mathbf{c}\varepsilon^{-p/(p-1)} + \log N), \quad \textit{if} \quad 1 < p < 2.$$

For $p = 1$ we have (13).

We want to prove that these upper bounds are optimal and start with a lower bound for $\mathrm{comp}_\varepsilon^{\mathrm{coin}}(S_N, \mathcal{B}(L_\infty^N))$. In the following we briefly write log instead of $\log_2$.

Lemma 2. *Let $N \in \mathbf{N}$ and $0 < \varepsilon < \sqrt{2}/2$. Then*

$$\mathrm{comp}_\varepsilon^{\mathrm{coin}}(S_N, \mathcal{B}(L_\infty^N)) \geq \frac{1}{2}\log N + \frac{1}{2}\log\bigl(2 - 2\sqrt{2}\,\varepsilon\bigr).$$

Proof. Let A be a restricted randomized algorithm with $e\bigl(S_N, A, \mathcal{B}(L_\infty^N)\bigr) \leq \varepsilon$. We set $f^{(1)} := (1, \ldots, 1) \in \mathcal{B}(L_\infty^N)$ and $k := \mathrm{cost}(A, f^{(1)})$. Without loss of generality we can assume that k is finite. We claim that

$$e\bigl(S_N, A, \mathcal{B}(L_\infty^N)\bigr) \geq \frac{1}{2}\sqrt{2}\left(1 - \frac{2^{2k-1}}{N}\right). \tag{18}$$

Assuming this claim for the moment, we obtain

$$\varepsilon \geq \frac{1}{2}\sqrt{2}\left(1 - \frac{2^{2k-1}}{N}\right).$$

It follows that

$$k \geq \frac{1}{2}\log N + \frac{1}{2}\log\bigl(2 - 2\sqrt{2}\,\varepsilon\bigr)$$

and

$$\mathrm{cost}\bigl(\mathcal{B}(L_\infty^N), A\bigr) \geq \frac{1}{2}\log N + \frac{1}{2}\log\bigl(2 - 2\sqrt{2}\,\varepsilon\bigr).$$

Hence (18) implies the lemma.

To prove (18), it suffices to consider the case $N > 2^{2k-1}$, since otherwise (18) is trivially true. Set

$$\Lambda := \{\omega \in \Omega : \mathrm{cost}(A, f^{(1)}, \omega) \leq 2k\}$$

so that, by Markov's inequality, $\mathbf{P}(\Lambda) \geq 1/2$. We will construct $f^{(2)} \in \mathcal{B}(L_\infty^N)$ such that $A_\omega(f^{(1)}) = A_\omega(f^{(2)})$ holds for all $\omega \in \Lambda$ but $|S_N(f^{(1)}) - S_N(f^{(2)})|$ is "large".

Let, for $\omega \in \Omega$, the set I_ω consist of those components of $f^{(1)}$ that are evaluated during the computational process for $A_\omega(f^{(1)})$. Hence, whenever algorithm A is applied to $f^{(1)}$ under use of $\omega \in \Lambda$ as realization of random bits, only components of $f^{(1)}$ belonging to $I := \bigcup_{\omega\in\Lambda} I_\omega$ are evaluated by A. It is not difficult to show that the cardinality $\mathrm{card}(I)$ of I is not greater than $\lfloor 2^{\lfloor 2k\rfloor - 1}\rfloor$. Since we assumed $N > 2^{2k-1}$, the element $f^{(2)} \in \mathcal{B}(L_\infty^N)$ given by $f^{(2)}(i) = 1$ if $i \in I$ and $f^{(2)}(i) = -1$ otherwise is well defined. Clearly, the definitions imply $A_\omega(f^{(1)}) = A_\omega(f^{(2)})$ for $\omega \in \Lambda$ and

$$|S_N(f^{(1)}) - S_N(f^{(2)})| = \frac{2N - 2\,\mathrm{card}(I)}{N} \geq \frac{2N - 2\cdot 2^{2k-1}}{N}\,.$$

It follows that

$$
\begin{aligned}
&e\big(S_N, A, \mathcal{B}(L_\infty^N)\big)^2 \\
&\geq \max_{i=1,2} \int_\Lambda e(S_N, A, f^{(i)}, \omega)^2 \, d\mathbf{P}(\omega) \\
&\geq \frac{1}{2} \int_\Lambda |S_N(f^{(1)}) - A_\omega(f^{(1)})|^2 \, d\mathbf{P}(\omega) \\
&\quad + \frac{1}{2} \int_\Lambda |S_N(f^{(2)}) - A_\omega(f^{(2)})|^2 \, d\mathbf{P}(\omega) \\
&= \frac{1}{4} \int_\Lambda 2|S_N(f^{(1)}) - A_\omega(f^{(1)})|^2 + 2|S_N(f^{(2)}) - A_\omega(f^{(1)})|^2 \, d\mathbf{P}(\omega).
\end{aligned}
$$

Using the relation $2a^2 + 2b^2 \geq (a+b)^2$ that holds for $a, b \in \mathbf{R}$, we obtain

$$
\begin{aligned}
&e\big(S_N, A, \mathcal{B}(L_\infty^N)\big)^2 \\
&\geq \frac{1}{4} \int_\Lambda \big(|S_N(f^{(1)}) - A_\omega(f^{(1)})| + |S_N(f^{(2)}) - A_\omega(f^{(1)})|\big)^2 \, d\mathbf{P}(\omega) \\
&\geq \frac{1}{4} \int_\Lambda \big(|S_N(f^{(1)}) - S_N(f^{(2)})|\big)^2 \, d\mathbf{P}(\omega) \\
&\geq \frac{1}{4} \cdot \left(\frac{2N - 2 \cdot 2^{2k-1}}{N}\right)^2 \cdot \frac{1}{2}.
\end{aligned}
$$

That is,

$$
e\big(S_N, A, \mathcal{B}(L_\infty^N)\big) \geq \frac{1}{2}\sqrt{2}\left(1 - \frac{2^{2k-1}}{N}\right).
$$

Now the proof of the lemma is complete. □

Since $\mathcal{B}(L_\infty^N) \subseteq \mathcal{B}(L_p^N)$, we easily derive the following result from Lemma 2.

Corollary 1. *Let $1 \leq p \leq \infty$. Then, for $0 < \varepsilon < \sqrt{2}/4$ and $N \in \mathbf{N}$,*

$$
\mathrm{comp}_\varepsilon^{\mathrm{coin}}(S_N, \mathcal{B}(L_p^N)) \geq \frac{1}{2} \log N \, .
$$

Combining the lower bound for $\mathrm{comp}_\varepsilon^{\mathrm{ran}}(S_N, \mathcal{B}(L_p^N))$ from Theorem 4 with that for $\mathrm{comp}_\varepsilon^{\mathrm{coin}}(S_N, \mathcal{B}(L_p^N))$ from Corollary 1, we obtain a lower bound for $\mathrm{comp}_\varepsilon^{\mathrm{coin}}(S_N, \mathcal{B}(L_p^N))$. This lower bound shows that the upper bound from Theorem 5 is optimal so that we obtain the exact order of complexity.

Theorem 6. *Let $1 < p \leq \infty$. Then, for $0 < \varepsilon < \sqrt{2}/4$ and $N \in \mathbf{N}$,*

$$
\mathrm{comp}_\varepsilon^{\mathrm{coin}}(S_N, \mathcal{B}(L_p^N)) \asymp \min(\mathbf{c}N, \mathbf{c}\varepsilon^{-\alpha} + \log N) \, ,
$$

where

$$
\alpha := \begin{cases} 2, & \text{if } \ 2 \leq p \leq \infty, \\ p/(p-1), & \text{if } \ 1 < p < 2. \end{cases}
$$

So far, we considered, for the upper bounds, algorithms where the number of function evaluations was fixed, the number of random bits used during the computational process, however, could vary. In what follows we look at even more special algorithms, namely at those where both the number of function evaluations and that of random bits are fixed.

For $m \in \mathbf{N}$, let $\mathbf{P}_m$ be the uniform distribution on $\Omega_m = \{0,1\}^m$, and let $\mathbf{E}_m$ denote the expectation with respect to $\mathbf{P}_m$.

A restricted randomized algorithm $A_{n,m} = \left(A^\omega_{n,m}\right)_{\omega \in \Omega_m}$ with n function evaluations and m random bits for the approximation of S has the form

$$A^\omega_{n,m}(f) = \varphi^\omega\left(f(i^\omega_1), \dots, f(i^\omega_n)\right)$$

where, for each $\omega \in \Omega_m$, the i^ω_k are arbitrary elements of D and $\varphi^\omega : \mathbf{R}^n \to \mathbf{R}$ is an arbitrary mapping. The quantity

$$e^{\mathrm{coin}}_{n,m}(S,F) = \inf_{A_{n,m}} e(S, A_{n,m}, F)\,,$$

where the infimum is taken over all restricted randomized algorithms $A_{n,m}$ with n function evaluations and m random bits, is called the randomized (n,m)-th minimal error. This corresponds to the randomized setting, just limited to the use of n function evaluations and m random bits. Hence

$$e^{\mathrm{coin}}_{n,m}(S,F) \geq e^{\mathrm{ran}}_n(S,F)\,. \tag{19}$$

We end the general discussion of restricted randomized algorithms with another simple observation.

Lemma 3. *It holds*

$$e^{\mathrm{coin}}_{n,m}(S,F) \geq e^{\mathrm{det}}_{n2^m}(S,F)\,.$$

Proof. Let $A_{n,m}$ be a restricted randomized algorithm with n function evaluations and m random bits. Then the deterministic algorithm

$$A(f) := \frac{1}{2^m} \sum_{\omega \in \Omega_m} A^\omega_{n,m}(f)$$

uses $n2^m$ function values. It follows

$$\begin{aligned} |S(f) - A(f)| &\leq \frac{1}{2^m} \sum_{\omega \in \Omega_m} \left|S(f) - A^\omega_{n,m}(f)\right| \\ &\leq \left(\frac{1}{2^m} \sum_{\omega \in \Omega_m} \left(S(f) - A^\omega_{n,m}(f)\right)^2\right)^{1/2} \\ &= e(S, A_{n,m}, f) \end{aligned}$$

for every $f \in F$. This implies the assertion. □

We now return to the summation problem and indicate a restricted randomized algorithm for the case where N is a power of 2, $N = 2^\ell$, say. This algorithm, we call it $\mathcal{A}_n^5$, uses n ($< N$) function values and 2ℓ random bits and has the form

$$(\mathcal{A}_n^5)^\omega(f) = \frac{1}{n}\sum_{k=1}^{n} f\big(i_k^\omega\big)\,,$$

where the i_k^ω are defined as follows. Let $a_0, a_1, \ldots, a_{N-1}$ be an arbitrary enumeration of the elements of the field with N elements. By using 2ℓ random bits, we take x^ω, $y^\omega \in \{a_0, a_1, \ldots, a_{N-1}\}$ independently according to the uniform distribution. Then we let i_k^ω be the (uniquely determined) number $z_k \in \{0, 1, \ldots, N-1\}$ that satisfies

$$a_{z_k} = x^\omega + a_k\, y^\omega\,.$$

Considerations similar to those that yield the formula of the individual error of algorithm $\mathcal{A}_n^3$, see Lemma 1, lead to

$$e(S_N, \mathcal{A}_n^5, f) = \frac{1}{\sqrt{n}}\left(\frac{1}{N}\sum_{i=0}^{N-1}\big(S(f) - f_i\big)^2\right)^{1/2}\,,$$

that is,

$$e(S_N, \mathcal{A}_n^5, f) = e(S_N, \mathcal{A}_n^1, f)\,.$$

Of course we can generalize $\mathcal{A}_n^5$ to arbitrary $N \in \mathbf{N}$ in the same way as we generalized algorithm $\mathcal{A}_n^4$, and we can define, for $1 \le p < 2$, the respective nonlinear modifications of $\mathcal{A}_n^5$. Consequently, we obtain the following result.

Theorem 7. *Let $1 \le p \le \infty$ and $\beta > 1$. Then, for $\beta n < N$,*

$$\begin{aligned} e^{\mathrm{coin}}_{n,2\lceil\log_2 N\rceil}\big(S_N, \mathcal{B}(L_p^N)\big) &\asymp e_n^{\mathrm{ran}}\big(S_N, \mathcal{B}(L_p^N)\big) \\ &\asymp n^{-1/2} \qquad (2 \le p \le \infty) \end{aligned}$$

$$\begin{aligned} e^{\mathrm{coin}}_{n,2\lceil\log_2 N\rceil}\big(S_N, \mathcal{B}(L_p^N)\big) &\asymp e_n^{\mathrm{ran}}\big(S_N, \mathcal{B}(L_p^N)\big) \\ &\asymp n^{-1+1/p} \qquad (1 \le p < 2)\,. \end{aligned}$$

It is remarkable that with the small number of $2\lceil\log_2 N\rceil$ random bits one can achieve the same (optimal) rate of convergence for the summation problem as with arbitrary random numbers.

3 The Integration Problem

Let $D = [0,1]^d$ be the d-dimensional unit cube, and let $C(D)$ denote the space of continuous functions on D, endowed with the supremum norm. For

$1 \le p \le \infty$, let $L_p(D)$ be the space of real-valued p-integrable functions, equipped with the norm

$$\|f\|_{L_p(D)} = \left(\int_D |f(t)|^p dt \right)^{1/p}$$

for $p < \infty$ and

$$\|f\|_{L_\infty(D)} = \text{ess sup}_{t \in D} |f(t)| .$$

Let, furthermore, $r \in \mathbf{N}$. The Sobolev space $W_p^r(D)$ consists of all functions $f \in L_p(D)$ such that for all multiindices $\alpha = (\alpha_1, \ldots, \alpha_d) \in \mathbf{N}_0^d$ with $|\alpha| := \sum_{j=1}^d \alpha_j \le r$, the generalized partial derivative $\partial^\alpha f$ belongs to $L_p(D)$. The norm on $W_p^r(D)$ is defined as

$$\|f\|_{W_p^r(D)} = \left(\sum_{|\alpha| \le r} \|\partial^\alpha f\|_{L_p(D)}^p \right)^{1/p} .$$

We assume $r \cdot p > d$. By the Sobolev embedding theorem, see Adams [1], or Triebel [25], functions from $W_p^r(D)$ are continuous on D so that function values are well defined. Let $\mathcal{B}(W_p^r(D))$ be the unit ball of $W_p^r(D)$, and let $I_d : W_p^r(D) \to \mathbf{R}$ be the integration operator

$$I_d(f) = \int_D f(t)\, dt .$$

We consider restricted randomized algorithms for the approximation of I_d. We shall reduce the study of the quantities $e_{n,m}^{\text{coin}}(I_d, \mathcal{B}(W_p^r(D)))$ to that of $e_{n,m}^{\text{coin}}(S_N, \mathcal{B}(L_p^N))$. We apply a discretization technique that has been developed in Heinrich [12]. We show here that it works also for the restricted randomized setting. For the convenience of the reader and for completeness we recall the key parts of the technique.

Let J be any quadrature rule on $C(D)$,

$$Jf = \sum_{j=0}^{\kappa-1} a_j f(t_j) \qquad (f \in C(D))$$

with $a_j \in \mathbf{R}$ and $t_j \in D$, such that J is exact on $\mathcal{P}_{r-1}(D)$, the space of polynomials on D of degree not exceeding $r-1$, that is,

$$Jf = I_d f \quad \text{for all} \quad f \in \mathcal{P}_{r-1}(D) . \tag{20}$$

For $d = 1$ one can take, e. g., Newton-Cotes formulas of appropriate degree and for $d > 1$ their tensor products. Let

$$D = \bigcup_{i=0}^{2^{dl}-1} D_{li}$$

be the partition of the unit cube D into 2^{dl} congruent cubes of disjoint interior, and let s_{li} denote the point of D_{li} with minimal coordinates. Finally, let $E_{li} : C(D) \to C(D)$ be the extension operator defined by

$$(E_{li}f)(s) = f(s_{li} + 2^{-l}s)$$

for $f \in C(D)$ and $s \in D$. Define, for $l \in \mathbf{N}_0$,

$$J_l f = 2^{-dl} \sum_{i=0}^{2^{dl}-1} J(E_{li}f) = 2^{-dl} \sum_{i=0}^{2^{dl}-1} \sum_{j=0}^{\kappa-1} a_j f(s_{li} + 2^{-l}t_j)\,,$$

which is the composed quadrature obtained by scaling J to the subcubes D_{li}. It is known that (20) implies

$$|I_d f - J_l f| \le c\, 2^{-rl} \|f\|_{W_p^r(D)} \quad (f \in W_p^r(D)) \tag{21}$$

(see, e. g., Heinrich [12] for a proof).

Now we are ready for the discretization process. Here is an outline of it. We shall approximate $I_d f$ by the quadrature $J_k f$ for some k, giving the desired precision, but having a number of nodes much larger than n. This J_k, in turn, will be split into the sum of a single quadrature J_{k_0}, with number of nodes of the order n, which we compute deterministically, and a hierarchy of (differences of) quadratures J_l' $(l = k_0, \ldots, k-1)$. It will be shown that the computation of the $J_l' f$ reduces to the computation of the mean of sequences with well-bounded $L_p^{N_l}$-norms for suitable N_l. This enables us to apply the results of Sect. 2.3 and approximate the means by algorithms with restricted randomization. Let us now give the details.

Define

$$\begin{aligned} J'f := (J_1 - J_0)f &= 2^{-d} \sum_{i=0}^{2^d-1} \sum_{j=0}^{\kappa-1} a_j f(s_{1,i} + 2^{-1}t_j) - \sum_{j=0}^{\kappa-1} a_j f(t_j) \\ &=: \sum_{j=0}^{\kappa'-1} a_j' f(t_j')\,, \end{aligned}$$

where

$$\kappa' \le \kappa(2^d + 1) \tag{22}$$

For $l \in \mathbf{N}_0$, set

$$\begin{aligned} J_{li}' f &= J'(E_{li}f) = \sum_{j=0}^{\kappa'-1} a_j' f(s_{li} + 2^{-l}t_j')\,, \\ J_l' &= 2^{-dl} \sum_{i=0}^{2^{dl}-1} J_{li}'\,. \end{aligned} \tag{23}$$

It is easily checked that

$$J_{l+1}f = 2^{-dl} \sum_{i=0}^{2^{dl}-1} J_1(E_{li}f)\,,$$

hence

$$\begin{aligned} J_{l+1}f - J_l f &= 2^{-dl} \sum_{i=0}^{2^{dl}-1} \left(J_1(E_{li}f) - J_0(E_{li}f)\right) \\ &= 2^{-dl} \sum_{i=0}^{2^{dl}-1} J'_{li}f = J'_l f\,, \end{aligned}$$

and therefore

$$J_k = J_{k_0} + \sum_{l=k_0}^{k-1} J'_l\,. \tag{24}$$

We have, by (23),

$$J'_l f = S_{N_l}\big((J'_{li}f)_{i=0}^{N_l-1}\big) = S_{N_l}(\Gamma_l f)\,, \tag{25}$$

where $N_l = 2^{dl}$ and $\Gamma_l : W_p^r(D) \to L_p^{N_l}$ is defined as

$$\Gamma_l f = (J'_{li}f)_{i=0}^{N_l-1}\,. \tag{26}$$

The estimate of the norms of these operators is crucial for the discretization technique. It is shown in Heinrich [12] that

$$2^{-dl} \sum_{i=0}^{2^{dl}-1} |J'_{li}f|^p \le c\, 2^{-prl} \|f\|^p_{W_p^r(D)}\,,$$

thus,

$$\|\Gamma_l\| \le c\, 2^{-rl}\,. \tag{27}$$

Lemma 4. *Let $1 \le p \le \infty$. There are constants $c_1, c_2 > 0$ such that for all $k_0 < k \in \mathbf{N}$, $m \in \mathbf{N}$, and $n_l \in \mathbf{N}$ $(l = k_0, \dots, k-1)$*

$$\begin{aligned} & c_1 \max_{k_0 \le l \le k-1} 2^{-rl} e^{\mathrm{coin}}_{n,m}\big(S_{N_l}, \mathcal{B}(L_p^{N_l})\big) \\ & \le e^{\mathrm{coin}}_{n,m}(I_d, \mathcal{B}(W_p^r(D))) \\ & \le c_2\, 2^{-rk} + c_2 \sum_{l=k_0}^{k-1} 2^{-rl} e^{\mathrm{coin}}_{n_l,m}\big(S_{N_l}, \mathcal{B}(L_p^{N_l})\big)\,, \end{aligned}$$

where n is defined as

$$n = \kappa 2^{dk_0} + \kappa' \sum_{l=k_0}^{k-1} n_l\,.$$

Proof. It follows readily from (24), the definitions, and the triangle inequality in $L_2(\Omega_m, \mathbf{P}_m)$ that

$$e^{\mathrm{coin}}_{n,m}(I_d, \mathcal{B}(W^r_p(D))) \le \sup_{f \in \mathcal{B}(W^r_p(D))} |I_d f - J_k f| + \sum_{l=k_0}^{k-1} e^{\mathrm{coin}}_{\kappa' n_l, m}(J'_l, \mathcal{B}(W^r_p(D)))\,.$$

Relation (27) and the structure of Γ, (25) and (26), imply

$$e^{\mathrm{coin}}_{\kappa' n_l, m}(J'_l, \mathcal{B}(W^r_p(D))) \le c\, 2^{-rl} e^{\mathrm{coin}}_{n_l, m}(S_{N_l}, \mathcal{B}(L^{N_l}_p))\,.$$

This together with (21) shows the upper bound. The lower bound follows in the usual way using bump functions. □

Theorem 8. *Let $r, d \in \mathbf{N}$ such that $r \cdot p > d$ and $1 \le p \le \infty$. Then*

$$\begin{aligned} e^{\mathrm{coin}}_{n,(2+d)\log_2 n}(I_d, \mathcal{B}(W^r_p(D))) &\asymp e^{\mathrm{ran}}_n(I_d, \mathcal{B}(W^r_p(D))) \\ &\asymp n^{-r/d-1/2} \qquad (2 \le p \le \infty) \end{aligned}$$

and

$$\begin{aligned} e^{\mathrm{coin}}_{n,(2+d)\log_2 n}(I_d, \mathcal{B}(W^r_p(D))) &\asymp e^{\mathrm{ran}}_n(I_d, \mathcal{B}(W^r_p(D))) \\ &\asymp n^{-r/d-1+1/p} \qquad (1 \le p < 2)\,. \end{aligned}$$

Proof. Since the order of convergence of $e^{\mathrm{ran}}_n(I_d, \mathcal{B}(W^r_p(D)))$ is well known, it suffices, by (19), to prove the upper bound for $e^{\mathrm{coin}}_{n,(2+d)\log_2 n}(I_d, \mathcal{B}(W^r_p(D)))$. We define, for $n \in \mathbf{N}$ sufficiently large,

$$k_0 := \lfloor \log_2(n/\kappa)/d \rfloor \qquad \text{and} \qquad k := \lceil (1 + d/(2r))k_0 \rceil$$

so that we have $k_0, k \in \mathbf{N}$ and $k_0 < k$. By our choice of k_0 and k, we obtain

$$2^{-d}\, n/\kappa \le 2^{dk_0} \le n/\kappa \tag{28}$$

and

$$2^{-kr} \asymp n^{-r/d-1/2}\,. \tag{29}$$

Let $0 < \delta < r$. We set, for $l = k_0, \ldots, k-1$,

$$n_l := \lceil 2^{dk_0 - \delta(l-k_0)} \rceil.$$

Using (22) and (28), we obtain,

$$\begin{aligned} \kappa\, 2^{dk_0} + \kappa' \sum_{l=k_0}^{k-1} n_l &\le \kappa\, n/\kappa + \kappa\,(2^d+1) \sum_{l=k_0}^{k-1} \left(2^{dk_0 - \delta(l-k_0)} + 1\right) \\ &= n + \kappa\,(2^d+1) \left(2^{dk_0} \sum_{l=0}^{k-1-k_0} \left(2^{-\delta}\right)^l + (k-k_0)\right) \\ &\le n + \kappa\,(2^d+1) \left(2^{dk_0}/\left(1-2^{-\delta}\right) + dk_0/(2r) + 1\right) \\ &\le n + \kappa\,(2^d+1) \left(2^{dk_0}/\left(1-2^{-\delta}\right) + 2 \cdot 2^{dk_0}\right) \\ &\le n + \kappa\,(2^d+1)\, n/\kappa \left(1/\left(1-2^{-\delta}\right) + 2\right). \end{aligned}$$

Hence, with $\widetilde{n} = \kappa 2^{dk_0} + \kappa' \sum_{l=k_0}^{k-1} n_l$,

$$\widetilde{n} \le c_1\, n \tag{30}$$

for some constant $c_1 \ge 1$. By Theorem 7 we have, for $l = k_0, \dots, k-1$,

$$e^{\rm coin}_{n_l, 2\lceil \log_2 N_{k-1}\rceil}\big(S_{N_l}, \mathcal{B}(L_p^{N_l})\big) \prec n_l^{-1/2} \qquad (2 \le p \le \infty)$$

and

$$e^{\rm coin}_{n_l, 2\lceil \log_2 N_{k-1}\rceil}\big(S_{N_l}, \mathcal{B}(L_p^{N_l})\big) \prec n_l^{-1+1/p} \qquad (1 \le p < 2)\,.$$

Taking into account

$$\begin{aligned} \lceil \log_2 N_{k-1}\rceil &= d(k-1) \\ &\le d(1+d/(2r))\lfloor d^{-1}\log_2(n/\kappa)\rfloor \\ &\le (1+d/2)\log_2 n\,, \end{aligned}$$

we arrive, with Lemma 4, (29) and after some straightforward calculations, at

$$e^{\rm coin}_{c_1 n, (2+d)\log_2 n}(I_d, \mathcal{B}(W_p^r(D))) \le c_2 n^{-r/d-1/2} \qquad (2 \le p \le \infty) \tag{31}$$

and

$$e^{\rm coin}_{c_1 n, (2+d)\log_2 n}(I_d, \mathcal{B}(W_p^r(D))) \le c_2 n^{-r/d-1+1/p} \qquad (1 \le p < 2)\,. \tag{32}$$

To conclude the proof, we use the monotonicity of the numbers $e^{\rm coin}_{n,m}$ with respect to n and m. Replacing n in (31) by $\lfloor n/c_1\rfloor$, we get (for sufficiently large n, and recalling that $c_1 \ge 1$)

$$\begin{aligned} e^{\rm coin}_{n,(2+d)\log_2 n}(I_d, \mathcal{B}(W_p^r(D))) &\le e^{\rm coin}_{c_1\lfloor n/c_1\rfloor, (2+d)\log_2\lfloor n/c_1\rfloor}(I_d, \mathcal{B}(W_p^r(D))) \\ &\le c_2 \lfloor n/c_1\rfloor^{-r/d-1/2} \le c_3 n^{-r/d-1/2} \end{aligned}$$

for $2 \le p \le \infty$. In the case $1 \le p < 2$ we argue similarly. □

Acknowledgement. The authors thank H. Woźniakowski for valuable discussions on the subject of this paper and for mentioning references [4, 5].

References

1. Adams, R.A.: Sobolev Spaces. Academic Press, New York (1975)
2. Bakhvalov, N.S.: On approximate computation of integrals. Vestnik Moskov. Gos. Univ. Ser. Math. Mech. Astron. Phys. Chem., **4**, 3–18 (1959) [In Russian]
3. Bakhvalov, N.S.: On the rate of convergence of indeterministic integration processes within the functional classes $W_p^{(l)}$. Theory Probab. Appl., **7**, 227 (1962)
4. Bakhvalov, N.S.: Optimal convergence bounds for quadrature processes and integration methods of Monte Carlo type for classes of functions. Ž. Vyčisl. Mat. i Mat. Fiz., **4**, suppl. 5–63 (1964) [In Russian]

5. Bakhvalov, N.S.: A lower bound for the randomness measure required in the use of the Monte-Carlo method. U.S.S.R. Comput. Math. Math. Phys. **5** (4), 252–257 (1965)
6. Bakhvalov, N.S.: On the optimality of linear methods for operator approximation in convex classes of functions. U.S.S.R. Comput. Math. Math. Phys., **11**, 244–249 (1971)
7. Blum, L., Cucker, F., Shub, M., Smale, S.: Complexity and Real Computation. Springer Verlag, New York (1998)
8. Chor, B., Goldreich, O.: On the power of two-point based sampling. J. Complexity, **5**, 96–106 (1989)
9. Goldreich, O., Wigderson, A.: Tiny families of functions with random properties: a quality-size trade-off for hashing. Random Structures and Algorithms, **11**, 315–343 (1997)
10. Hardy, G.H., Wright, E.M.: An Introduction to the Theory of Numbers. Third edition. Clarendon Press, Oxford (1954)
11. Heinrich, S.: Random approximation in numerical analysis. In: Bierstedt, K.D., Pietsch, A., Ruess, W.M., Vogt, D. (eds) Functional Analysis. Proceedings of the Essen Conference, held in Essen, Germany, November 24–30, 1991. Dekker, New York (1994)
12. Heinrich, S.: Quantum integration in Sobolev classes. J. Complexity **19**, 19–42 (2003). See also http://arXiv.org/abs/quant-ph/0112153.
13. Joffe, A.: On a set of almost deterministic k-independent random variables. Ann. Probab., **2**, 161–162 (1974)
14. Mathé, P.: Random approximation of finite sums. Preprint 11. Institute for Applied Analysis and Stochastics, Berlin (1992)
15. Mathé, P.: The optimal error of Monte Carlo integration. J. Complexity, **11**, 394–415 (1995)
16. Motwani, R., Raghavan, P.: Randomized Algorithms. Cambridge University Press, Cambridge (1995)
17. Novak, E.: Eingeschränkte Monte Carlo-Verfahren zur numerischen Integration. In: Grossmann, W., Pflug, G.C., Vincze, I., Wertz, W. (eds) Mathematical statistics and applications. Proceedings of the 4th Pannonian Symposium on Mathematical Statistics, Bad Tatzmannsdorf, Austria, September 4–10, 1983, Volume B. D. Reidel Publishing Company, Dordrecht (1985)
18. Novak, E.: Deterministic and Stochastic Error Bounds in Numerical Analysis. Lecture Notes in Mathematics **1349**. Springer, Berlin Heidelberg New York (1988)
19. Novak, E.: The real number model in numerical analysis. J. Complexity, **11**, 57–73 (1995)
20. Novak, E.: On the power of adaption. J. Complexity, **12**, 199–237 (1996)
21. Novak, E.: Quantum complexity of integration. J. Complexity, **17**, 2–16 (2001). See also http://arXiv.org/abs/quant-ph/0008124
22. Sugita, H., Takanobu, S.: Random Weyl sampling for robust numerical integration of complicated functions. Monte Carlo Methods Appl., **6**, 27–48 (2000)
23. Traub, J.F., Wasilkowski, G.W., Woźniakowski, H.: Information-Based Complexity. Academic Press, New York (1988)
24. Traub, J.F., Woźniakowski, H.: The Monte Carlo algorithm with a pseudorandom generator. Math. Comp., **58**, 323–339 (1992)
25. Triebel, H.: Interpolation Theory, Function Spaces, Differential Operators. Second edition. Barth, Leipzig (1995)