Numerical integration for complicated functions and random sampling

Hiroshi SUGITA

1 Introduction

In general, to each smoothness class of integrands, there corresponds an appropriate numerical integration method, and the error is estimated in terms of the norm of the integrand and the number of sampling points. The most widely applicable numerical integration method by means of deterministic sampling points is so-called quasi-Monte Carlo method, which is used for integrands of bounded variations(cf.(3) below).

But if the integrand is more complicated, for example, if it has a very big total variation relative to its integral or if it is not of bounded variation, quasi-Monte Carlo method may not work for it. In such a case, numerical integration by means of random sampling — Monte-Carlo integration — is indispensable.

Random variables are usually very complicated when they are realized on a concrete probability space. So numerical integration for complicated functions is needed to calculate their means(expectations). But of course, there are many random variables which are not so complicated, for which deterministic numerical integration methods work. If we need random sampling for a random variable, it is not because it is governed by chance, but because it is complicated. In this sense, we gave this article the title 'Numerical integration for complicated functions and . . . ', although we put 'numerical integration for random variables' in mind.

Random sampling for numerical integration by computer has begun by von Neumann and Ulam, as soon as electronic computer was born. It was also the birth of the problem 'How do we realize random sampling by computer?'

In 1960's, Kolmogorov, Chaitin and others began the theory of random numbers ([8, 16]), known as 'Kolmogorov's complexity theory', by using the theory of computation, which was in the cradle at that time. When Martin-Löf proved that a $\{0,1\}$-sequence is random if and only if it is accepted by the so-called universal test, which assymptotically dominates any test, the theory of random numbers was established([20]). The random numbers however are quite useless in practice, because a $\{0,1\}$-sequence is called random if there is no shorter algorithm to generate it than the algorithm that writes down the sequence itself. Moreover, since the universal test was constructed to keep the asymptotic consistency, even if we are given random numbers, they do not necessarily look random in practical scale.

Anyhow, to meet practical demand, many 'pseudo-random generators' have been developed without a reasonable theory, and they have been successful at least in engineering([18, 21, 22, 35]).

Twenty years after Kolmogorov's theory, a clear "definition" of pseudo-random generator was presented in 1980's in the theory of cryptography([3, 4, 37]). It was based on the theory of computational complexity, which was in a sense a practical refinement of Kolmogorov's complexity theory. According to the "definition", pseudo-random generation is a technique to let small randomness look large randomness. Just like gilding, the purpose of pseudo-random generation is to paste little precious resource (= small randomness) onto surface (= finite-dimensional distributions which are tested by feasible procedures). We do not mind what one cannot see from the outside (= finite-dimensional distributions which cannot be tested by any feasible procedure). A pseudo-random generator which achieves this purpose is called 'secure'(Definition 4.1).

The reason why we wrote "definition" with double quotation marks is because there is no proof of the existence of mathematical object that satisfies the requirement of the "definition", that is, secure pseudo-random generator. It is a difficult problem involving the $\mathbf{P} \neq \mathbf{NP}$ conjecture. Nevertheless, it is a big progress that the essential problem of pseudo-random generation has been made clear.

But if we restrict the use of pseudo-random generator, there is a possibility to 'let small randomness work just like large randomness'. For example, as for Monte Carlo integration, this has been already achieved(§ 4.3).

While random sampling is needed for numerical integration of complicated functions, it is often the case that sample sequences look very random when we apply quasi-Monte Carlo method to complicated functions([10, 11, 25, 31, 32]). A pseudo-random generator has been constructed by using this phenomenon([25]), whose security we discuss in § 5.2.

Random sampling by computer, whose main user is engaged in applied sciences, has been very often discussed by intuition, and spread without rigor. Unfortunately, the fact is that even among the professional researchers, no consensus on it has been yet established. Thus what follows is the author's proposal as a researcher of probability theory.

2 Lebesgue space and coin tossing process

First, we introduce some symbols and notations.

Definition 2.1 (i) Let $\mathbb{T}^1$ be 1 dimensional torus (a group $[0,1)$ with addition $x + y \bmod 1$), $\mathcal{B}$ be the Borel σ field over $\mathbb{T}^1$, and $\mathbb{P}$ be the Lebesgue measure. We call the probability space $(\mathbb{T}^1, \mathcal{B}, \mathbb{P})$ the Lebesgue probability space.
(ii) For each $m \in \mathbb{N}$, let $d_m(x) \in \{0,1\}$ denote the m-th digit of $x \in \mathbb{T}^1$ in its dyadic expansion, that is,

$$d_1(x) = \mathbf{1}_{[1/2,1)}(x), \quad d_m(x) = d_1(2^{m-1}x), \quad x \in \mathbb{T}^1.$$

(iii) For each $m \in \mathbb{N}$, we put $\Sigma^m := \{\, i/2^m \mid i = 0, \ldots, 2^m - 1 \,\} \subset \mathbb{T}^1$, $\mathcal{B}_m := \sigma\{\, [a,b) \mid a, b \in \Sigma^m,\ a < b \,\}$, and for $x \in \mathbb{T}^1$,

$$\lfloor x \rfloor_m := \lfloor 2^m x \rfloor / 2^m = \sum_{i=1}^{m} 2^{-i} d_i(x) \in \Sigma^m.$$

By convention, we define $\lfloor x \rfloor_\infty := x$.
(iv) For each $m \in \mathbb{N}$, let P_m denote the uniform probability measure on the measurable space $(\Sigma^m, 2^{\Sigma^m})$.

The sequence of functions $\{d_m\}_{m=1}^\infty$ is a fair coin tossing process on the Lebesgue probability space([5, 6]). $\mathcal{B}_m$ is the set of events which are determined by at most m coin tosses. f is $\mathcal{B}_m$-measurable if and only if $f(x) \equiv f(\lfloor x \rfloor_m)$. The probability space $(\Sigma^m, 2^{\Sigma^m}, P_m)$ describes everything about m coin tosses. The probability space $(\mathbb{T}^1, \mathcal{B}_m, \mathbb{P})$ is isomorphic to $(\Sigma^m, 2^{\Sigma^m}, P_m)$ by a mapping $\lfloor \cdot \rfloor_m : \mathbb{T}^1 \to \Sigma^m$. If f is $\mathcal{B}_m$-measurable, we say 'f has (at most) m-bit randomness'.

The mean(expectation) of a random variable f on the Lebesgue probability space is denoted by $I[f]$, that is,

$$I[f] := \int_{\mathbb{T}^1} f(y) dy.$$

For any random variable whose defining probability space is not specifically shown, the mean shall be denoted by $\mathbf{E}$.

3 Numerical integration for complicated functions

3.1 A numerical example

Let us make a detour to present a numerical example for the later explanation.

On the Lebesgue probability space, we define two random variables with parameter $m \in \mathbb{N}$: For $x \in \mathbb{T}^1$,

$$S_m(x) := \sum_{i=1}^{m} d_i(x), \tag{1}$$

$$f_m(x) := \mathbf{1}_{\{S_{2m-1}(\cdot) \le m-1\}}(x) = \begin{cases} 1, & (S_{2m-1}(x) \le m-1) \\ 0, & (S_{2m-1}(x) \ge m) \end{cases}. \tag{2}$$

The former returns the number of Heads ($d_i(x) = 1$) out of m tosses, while the latter returns 1 or 0 according with whether or not the number of Heads is at most $(m-1)$ out of $(2m-1)$ tosses.

Now let us consider to calculate the integral of f_{50}. Since Head and Tail come out with equal probabilities, we know $I[f_{50}] = \mathbb{P}(S_{99} \le 49) = 1/2$. Of course, it is most desirable that we can calculate the integral by theoretical analysis in this way.

The second most desirable is to be able to compute an approximated integral value by a deterministic method with a meaningful error estimate. Although f_{50} is a rather simple random variable as a probabilistic object, it has countless discontinuous points as a function on $\mathbb{T}^1$. So we can imagine that no deterministic numerical integration method works for it.

But as an experiment, let us apply quasi-Monte Carlo methods to f_{50}. A quasi-Monte Carlo method is a deterministic sampling method by means of a low discrepancy sequence. A $[0,1) = \mathbb{T}^1$-valued sequence $\{x_n\}_{n=1}^{\infty}$ is said to be of low discrepancy, if for any function f on $\mathbb{T}^1$ of bounded variation, we have

$$\left| \frac{1}{N} \sum_{n=1}^{N} f(x_n) - I[f] \right| \le c(N) ||f||_{BV} \times \frac{\log N}{N}, \quad N \in \mathbb{N}, \tag{3}$$

where $||f||_{BV}$ is the total variation of f and $C(N) > 0$ is independent of f([17, 22]). For example, the Weyl sequence with irrational parameter $\alpha = (\sqrt{5}-1)/2$,

$$x_n := (n\alpha) \bmod 1, \quad n = 1, 2, \ldots, \tag{4}$$

satisfies (3) for $c(N) = (3/\log N) + (1/\log \xi) + (1/\log 2)$, $\xi = (1+\sqrt{5})/2$ ([17]).

Note that $||f_{50}||_{BV}$ is so huge that the error estimate (3) has no practical meaning. But, anyway, we calculated the absolute error $|N^{-1} \sum_{n=1}^{N} f_{50}(x_n) - (1/2)|$, where x_n is defined by (4) and $N = 10^3, 10^4, \ldots, 10^7$. The result is shown in Fig.1. The numerical integration seems to be successful. As the declined line is of slope $-1/2$, the convergence rate is approximately $O(N^{-1/2})$.

Using the *van der Corput sequence* $\{v_n\}_{n=1}^{\infty}$, another well-known low discrepancy sequence, let us try the same calculation. Here v_n is defined by

$$v_n := \sum_{i=1}^{\infty} d_{-i}(n-1) 2^{-i}, \qquad n = 1, 2, \ldots, \tag{5}$$

where $d_{-i}(n)$ stands for the i-th bit of $n = 0, 1, 2, \ldots$, in its binary expansion (i.e., $n = \sum_{i=1}^{\infty} d_{-i}(n) 2^{i-1}$). To put it concretely,

$$\{v_n\}_{n=1}^{\infty} = \left\{ 0, \frac{1}{2}, \frac{1}{4}, \frac{3}{4}, \frac{1}{8}, \frac{5}{8}, \frac{3}{8}, \frac{7}{8}, \frac{1}{16}, \frac{9}{16}, \frac{5}{16}, \frac{13}{16}, \ldots \right\}. \tag{6}$$

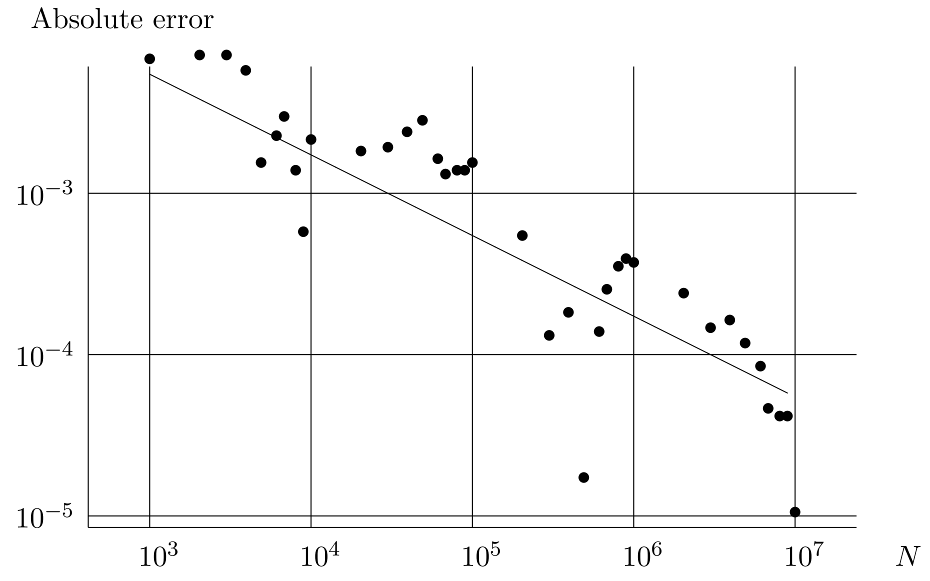

Figure 1: Absolute error of quasi-Monte Carlo method by the Weyl sequence (4)(log-log graph)

This time, letting $x_n = v_n$, (3) holds with $c(N) = \log(N+1)/(\log 2 \cdot \log N)$. In comparison of $c(N)$, the van der Corput sequence is better than the Weyl sequence. But if $N < 2^{50} = 1.13 \times 10^{15}$, we have $N^{-1}\sum_{n=1}^{N} f_{50}(v_n) = 1$, which means that $\{v_n\}_{n=1}^{\infty}$ does not work for the integration of f_{50} in practice.

3.2 Why deterministic methods do not work

What numerical integration method is applicable to the set of $\mathcal{B}_m$-measurable functions? We have $I[f] = 2^{-m}\sum_{y\in\Sigma^m} f(y)$, for $\mathcal{B}_m$-measurable f. If 2^m is small enough, it is easy to compute the right-hand side's finite sum. The problem is how we compute it when 2^m is so large that we cannot compute the finite sum. What we can do in this case is to compute an approximated value of $I[f]$ from the values $\{f(x_n)\}_{n=1}^{N}$ $(N \ll 2^m)$ of f at sample points $\{x_n\}_{n=1}^{N} \subset \Sigma^m$. In such a situation, as we will explain soon, any deterministic numerical integration method does not work.

Let us consider a numerical integration method of the type

$$I\left(f; \mathbf{c}, \mathbf{x}\right) := \sum_{n=1}^{N} c_n f(x_n), \tag{7}$$

where $\mathbf{c} := \{c_n\}_{n=1}^{N}$ is a coefficient vector which satisfies $\sum_{n=1}^{N} c_n = 1$. Let $L^2(\mathcal{B}_m)$ denote the set of all $\mathcal{B}_m$-measurable real valued functions with the usual inner product $\langle f, g\rangle := I[fg]$. The norm is given by $||f|| := \langle f, f\rangle^{1/2}$ as usual.

Now, since $I\left(f; \mathbf{c}, \mathbf{x}\right)$ is linear in $f \in L^2(\mathcal{B}_m)$, there exists a unique $g_{\mathbf{c},\mathbf{x}} \in L^2(\mathcal{B}_m)$ such that $I\left(f; \mathbf{c}, \mathbf{x}\right) = \langle g_{\mathbf{c},\mathbf{x}}, f\rangle$ for each $f \in L^2(\mathcal{B}_m)$. Indeed, it is given by

$$g_{\mathbf{c},\mathbf{x}}(y) := 2^m \sum_{n=1}^{N} c_n \mathbf{1}_{[x_n, x_n+2^{-m})}(y), \quad y \in \mathbb{T}^1. \tag{8}$$

Consequently, we have

$$I\left(f; \mathbf{c}, \mathbf{x}\right) - I[f] = \langle g_{\mathbf{c},\mathbf{x}} - 1, f\rangle.$$

This means that in order to make the error $|I(f;\mathbf{c},\mathbf{x})-I[f]|$ small, we must take $\mathbf{c},\mathbf{x}$ so that f and $g_{\mathbf{c},\mathbf{x}}-1$ is almost orthogonal. But can we take $\mathbf{c},\mathbf{x}$ so that $g_{\mathbf{c},\mathbf{x}}-1$ is orthogonal to all $f\in L^2(\mathcal{B}_m)$? Of course, we cannot, because $g_{\mathbf{c},\mathbf{x}}-1\in L^2(\mathcal{B}_m)$ has same direction with itself.

Indeed, noting $\langle g_{\mathbf{c},\mathbf{x}},1\rangle = 1$, $||g_{\mathbf{c},\mathbf{x}}||^2 \ge 2^m\sum_{n=1}^N c_n^2$ (if x_n's are all distinct then '=' holds), and that $\sum_{n=1}^N c_n^2 \ge 1/N$ (if $c_n\equiv 1/N$ then '=' holds), we have

$$\begin{aligned} I(g_{\mathbf{c},\mathbf{x}}-1;\mathbf{c},\mathbf{x})-I[g_{\mathbf{c},\mathbf{x}}-1] &= \langle g_{\mathbf{c},\mathbf{x}}-1, g_{\mathbf{c},\mathbf{x}}-1\rangle = ||g_{\mathbf{c},\mathbf{x}}||^2-1 \\ &\ge 2^m\sum_{n=1}^N c_n^2 - 1 \ge \frac{2^m}{N}-1, \end{aligned} \tag{9}$$

which becomes huge if $N \ll 2^m$. In short, the sampling method by means of $\mathbf{c},\mathbf{x}$ yields a tremendous error if it is applied to the function $g_{\mathbf{c},\mathbf{x}}-1$ corresponding to itself.

This argument is quite trivial, but the theoretical obstacle in numerical integration of complicated functions by means of finite deterministic sampling points consists in this 'self-duality'. The failure of numerical integration of f_{50} by means of van der Corput sequence $\{v_n\}_{n=1}^\infty$ shows that this kind of matter can occur in practice.

Remark 3.1 If our considering class L of integrands forms a thin set in $L^2(\mathcal{B}_m)$, like a linear subspace, then we may able to take $\mathbf{c},\mathbf{x}$ so that $g_{\mathbf{c},\mathbf{x}}-1$ can be almost orthogonal to L. Indeed, the trapezoidal rule as well as Simpson's rule has the form (7), and the corresponding smooth class of integrands (discretized into 2^{-m} precision) to the rule must be a thin set in $L^2(\mathcal{B}_m)$.

3.3 Scenario of Monte Carlo integration

Monte Carlo integration is the general term for numerical integration methods which use random variables as the sampling points. What follows is a summary of its idea[1)].

Purpose

Let $\tilde{f}$ be a random variable whose mean we wish to know. First, we construct a random variable f on $(\mathbb{T}^1,\mathcal{B},\mathbb{P})$ which has the same (or almost same) distribution as $\tilde{f}$. Now our aim is to compute $I[f]$ numerically instead of $\mathbf{E}[\tilde{f}]$. Next, we construct another random variable X also on $(\mathbb{T}^1,\mathcal{B},\mathbb{P})$ which distributes near around $I[f]$. In the present article, assuming $f\in L^2[0,1)$, we consider to construct X for any error bound $\varepsilon>0$ so that the mean square error is less than ε, i.e.,

$$I\left[|X-I[f]|^2\right] < \varepsilon. \tag{10}$$

Then Chebyshev's inequality shows

$$\mathbb{P}(|X-I[f]|>\delta) < \varepsilon/\delta^2. \tag{11}$$

Here note that f and X must be either of finite randomness or simulatable(see § 3.7).

When f and X satisfying all requirements above are constructed, the theoretical purpose is achieved.

Random sampling

In order to obtain an approximated value of $I[f]$, we evaluate $X(x)$ for an $x\in\mathbb{T}^1$. Of course, since we do not know in advance which x gives a good approximated value — if we knew, there would be a deterministic numerical integration method —, we are not sure to obtain a good approximation. Choosing an x is a kind of gambles, as the term 'Monte

Carlo' — famous city for casinos — indicates. The executer of the calculation, whom we shall call Alice below, has the freedom to choose an $x \in \mathbb{T}^1$ and simultaneously bears the responsibility for the result. Then (10) and (11) are considered to be the estimation of her risk. We call such a method of problem solution *random sampling.* Note that we should not mind the adjective 'random' so much. For example, the mathematical content of (11) is, if X is $\mathcal{B}_M$-measurable, that the number of $x \in \Sigma^M$ that satisfies $|X(x) - I[f]| > \delta$ is less than $2^M \varepsilon/\delta^2$, and nothing else.

Pseudo-random generator

However, the randomness of $X(= M)$ is very often extraordinarily large. In other words, to evaluate X, Alice needs to input an extraordinarily long binary string $x \in \Sigma^M$ into X. Then a pseudo-random generator $g : \Sigma^n \to \Sigma^M$, $n \ll M$, takes over her job. Namely, instead of a long binary string x, Alice chooses a short binary string $\omega \in \Sigma^n$ as an input to g. The output $g(\omega) \in \Sigma^M$ is called *pseudo-random bits* and ω is called the *seed.* Then she computes $X(g(\omega))$. Thus she actually has only freedom to choose ω but not x.

Now, how shall we estimate her risk? Since we do not know again which ω to choose, it is natural to assume that ω is a random variable uniformly distributed in Σ^n. Although, in general, it is almost impossible to know the exact distribution of $X(g(\omega))$ under P_n, Alice (usually unconsciously) takes it for granted that almost the same risk estimate as (11) holds for $X(g(\omega))$, i.e.,

$$P_n\left(|X(g(\omega)) - I[f]| > \delta\right) < \varepsilon'/\delta^2,$$

where ε' may be slightly bigger than ε. In other words, to meet Alice's expectation, the pseudo-random generator g should have the following property: The distribution of $X(g(\omega_n))$ under P_n is close to that of $X(x)$.

3.4 i.i.d.-sampling

Let $f \in L^2(\mathcal{B}_m)$ be our integrand, let $\{Z_n\}_{n=1}^N$ be an i.i.d.(= independently identically distributed) random variables each of which is uniformly distributed in Σ^m, and let

$$I_{\mathrm{iid}}(f; \{Z_n\}_{n=1}^N) := I(f; \{1/N\}_{n=1}^N, \{Z_n\}_{n=1}^N) = \frac{1}{N}\sum_{n=1}^N f(Z_n). \tag{12}$$

The random sampling that estimates $I[f]$ by means of $I_{\mathrm{iid}}(f; \{Z_n\}_{n=1}^N)$ is called *i.i.d.-sampling*, which is the simplest and most used random sampling.

As is well-known, the mean square error is estimated as

$$I\left[\left|I_{\mathrm{iid}}(f; \{Z_n\}_{n=1}^N) - I[f]\right|^2\right] = \frac{1}{N}\mathbf{Var}[f], \tag{13}$$

where $\mathbf{Var}[f] := I\left[|f - I[f]|^2\right]$ is the variance of f. It follows from Chebyshev's inequality that

$$\mathbb{P}\left(\left|I_{\mathrm{iid}}(f; \{Z_n\}_{n=1}^N) - I[f]\right| > \delta\right) \le \frac{\mathbf{Var}[f]}{N\delta^2}, \quad \delta > 0. \tag{14}$$

If N is large enough, the distribution of (12) can be approximated by some normal distribution, so that the error estimate (14) will be much more refined.

The i.i.d. random variables $\{Z_n\}_{n=1}^N$ can be realized on the Lebesgue probability space as $Z_n = Z_n(x) := \sum_{i=1}^m 2^{-i} d_{(n-1)m+i}(x)$, $x \in \mathbb{T}^1$. Then we see that $X(x) := I_{\mathrm{iid}}(f; \{Z_n(x)\}_{n=1}^N)$ is $\mathcal{B}_{Nm}$-measurable. Thus when $1 \ll N$, the randomness of X is much larger than that of f.

3.5 An inequality for general random sampling

Instead of a clear formula like (13), an inequality holds for general random sampling methods.

Theorem 3.1 (cf. [26]) *Let $\{1, \psi_1, \dots \psi_{2^m-1}\}$ be an orthonormal basis of $L^2(\mathcal{B}_m)$. Then for any random variables $\mathbf{X} := \{X_n\}_{n=1}^N \subset \mathbb{T}^1$, $1 \le N \le 2^m$, the following inequality holds.*

$$\sum_{l=1}^{2^m-1} \mathbf{E}\left[\,|I(\psi_l; \mathbf{c}, \mathbf{X})\,|^2\right] \ge \frac{2^m}{N} - 1. \tag{15}$$

If $c_n \equiv 1/N$ and $\{\lfloor X_n \rfloor_m\}_{n=1}^N$ are all distinct with probability 1, then (15) becomes an equality.

Proof. As a special case of the assertion, even if $\mathbf{X}$ is deterministic, (15) should be valid. On the other hand, if (15) is valid for each deterministic $\mathbf{X}$, then it is valid for any random $\mathbf{X}$. Thus we will show it for any deterministic sequence $\{x_n\}_{n=1}^N$.

We may assume $\mathbf{x} := \{x_n\}_{n=1}^N \subset \Sigma^m$. To this sequence, there corresponds $g_{\mathbf{c},\mathbf{x}} \in L^2(\mathcal{B}_m)$ via (8). Parseval's identity (or Pythagoras' theorem) implies that

$$||g_{\mathbf{c},\mathbf{x}}||^2 = \langle g_{\mathbf{c},\mathbf{x}}, 1\rangle^2 + \sum_{l=1}^{2^m-1} \langle g_{\mathbf{c},\mathbf{x}}, \psi_l\rangle^2. \tag{16}$$

It follows form $\langle g_{\mathbf{c},\mathbf{x}}, 1\rangle^2 = 1$, (9), and (16) that

$$\frac{2^m}{N} \le 1 + \sum_{l=1}^{2^m-1} |I(\psi_l; \mathbf{c}, \mathbf{x})|^2,$$

which is nothing but (15) for $X_n \equiv x_n$. □

If a sampling method — deterministic or random — is very efficient for a certain class of good integrands, we had better not use it for integrands which are not in the class, because there must exist a bad integrand for which the error becomes larger than that of i.i.d.-sampling so as to fulfill the inequality (15). Thus, this sampling provides a "high return and high risk"-method.

Conversely, i.i.d.-sampling provides the most "low risk and low return"-method. Let us call the supremum of the normalized mean square error

$$R(\mathbf{c}, \mathbf{X}) := \sup_{f \in L^2(\mathcal{B}_m),\, \mathbf{Var}[f]=1} \mathbf{E}\left[|I(f; \mathbf{c}, \mathbf{X}) - I[f]|^2\right] \tag{17}$$

the maximum risk of the random sampling by means of $\mathbf{c}, \mathbf{X}$. The maximum risk of i.i.d.-sampling is $R(\{1/N\}_{n=1}^N, \{Z_n\}_{n=1}^N) = 1/N$ by (13). For a general random sampling, dividing both sides of (15) by $2^m - 1$, we get

$$R(\mathbf{c}, \mathbf{X}) \ge \left(\frac{2^m}{2^m-1}\frac{1}{N} - \frac{1}{2^m-1}\right) \sim \frac{1}{N}, \qquad N \ll 2^m. \tag{18}$$

Thus, when $N \ll 2^m$, we can say that i.i.d.-sampling has the property that its maximum risk is almost smallest among those of all random sampling methods.

3.6 Reduction of randomness

The formula (13) for i.i.d.-sampling is a consequence of uncorrelatedness ($I[(f(Z_n) - I[f])(f(Z_{n'}) - I[f])] = 0$ if $n \neq n'$) rather than independence. We can therefore obtain (13) for uniformly distributed *pairwise independent* random variables instead of i.i.d. random variables.

Example 3.1 (cf. [19]) Identify Σ^m with $\{0,1\}^m$ through binary expansion and further with the Galois field $\mathrm{GF}(2^m)$. Let $\Omega := \mathrm{GF}(2^m) \times \mathrm{GF}(2^m)$ and let μ be the uniform probability measure on Ω. We define random variables $\{X_n\}_{n \in \mathrm{GF}(2^m)}$ on the probability space $(\Omega, 2^\Omega, \mu)$ by

$$X_n(\omega) := x + n\alpha, \quad \omega = (x, \alpha) \in \Omega, \quad n \in \mathrm{GF}(2^m).$$

Then each X_n distributes uniformly in $\mathrm{GF}(2^m)$, and if $n \neq n'$, X_n and $X_{n'}$ are independent.

The assertion of Example 3.1 follows from the fact that the following system of linear equations

$$\begin{cases} x + n\alpha &= a \\ x + n'\alpha &= a' \end{cases}$$

has a unique solution $(x, \alpha) = \omega_0 = (x_0, \alpha_0) \in \Omega$, and hence

$$\mu(X_n = a, X_{n'} = a') = \mu(\{\omega_0\}) = 2^{-2m} = \mu(X_n = a)\mu(X_{n'} = a').$$

To assure that two Σ^m-valued random variables X_n and $X_{n'}$ are independent, we need at least $2m$ bit randomness. Consequently, it is impossible to construct Σ^m-valued pairwise independent random variables with randomness less than Example 3.1. However, since the operations in $\mathrm{GF}(2^m)$ are in fact polynomial operations, if m is a little bit big, it is not easy to generate sampling points X_n quickly. Hence, these random variables are not practical for numerical calculus.

There are other methods for generation of pairwise independent random variables([2, 9, 12, 14]). Among them, the following example is very suited for numerical calculation, although it is not of least randomness.

Example 3.2 (*Random Weyl sampling*(RWS)[26, 33]) Let $\Omega := \Sigma^{m+j} \times \Sigma^{m+j}$ and $\mu := P_{m+j} \otimes P_{m+j}$(= the uniform probability measure on Ω). We define Σ^m-valued random variables $\{X_n\}_{n=1}^{2^j}$ on the probability space $(\Omega, 2^\Omega, \mu)$ by

$$X_n(\omega) := \lfloor x + n\alpha \rfloor_m, \quad \omega = (x, \alpha) \in \Omega, \quad n = 1, 2, 3, \ldots, 2^j.$$

Then under μ, each X_n distributes uniformly in Σ^m, and X_n and $X_{n'}$ are independent if $1 \leq n \neq n' \leq 2^j$. In particular, $\{f(X_n)\}_{n=1}^{2^j}$ are pairwise independent having the same distribution as f.

As an example, let us consider numerical integration of f_{50} defined by (2) again. If we apply i.i.d.-sampling to f_{50} with sample size $N = 10^7$, we need $99 \times 10^7 \sim 10^9$ random bits, while if we apply RWS to it with the same sample size, we need only $\lceil 99 + \log_2 10^7 \rceil \times 2 = 246$ random bits. Since a 246 bit string is easy to type, we need no pseudo-random generator in the latter case.

Of course, it often happens that even if we apply RWS we still need to input such a long binary string that we cannot type it. Then we use a pseudo-random generator. In this case, the drastic reduction of randomness of RWS brings us the following advantages:

(a) RWS is very insensitive to the quality of pseudo-random generator (§ 4 of [33]).

(b) RWS is very insensitive to the speed of pseudo-random generator. Consequently, we can use a slow but precise pseudo-random generator (such as secure one, see § 4.2). In this case, the random sampling becomes very reliable.

Remark 3.2 In § 3.1, we saw that quasi-Monte Carlo method by means of the Weyl sequence (4) worked well for the integration of f_{50}. This can be interpreted as follows: Regarding that what we did was RWS with $(x, \alpha) = (0, \lfloor(\sqrt{5}-1)/2\rfloor_{123}) \in \Sigma^{123} \times \Sigma^{123}$, we can say such good luck takes place very often.

3.7 Simulatable random variables and their numerical integration

By finiteness of computational resources(memory and time), we can deal with only random variables of finite randomness, i.e., $\mathcal{B}_m$-measurable function for some $m \in \mathbb{N}$. But the following class of random variables of infinite randomness (not $\mathcal{B}_m$-measurable for any $m \in \mathbb{N}$) can be dealt with rigorously with high probability.

Example 3.3 (Hitting time) A random variable σ on $(\mathbb{T}^1, \mathcal{B}, \mathbb{P})$ defined by

$$\sigma(x) := \inf\{n \geq 1 \mid d_1(x) + d_2(x) + \cdots + d_n(x) = 5\}, \quad x \in \mathbb{T}^1,$$

describes the time when the 5-th Head comes out in sequential coin tosses ($\inf \emptyset := \infty$). Clearly, σ is unbounded and of infinite randomness. But, when the 5-th Head comes out, the rest of coin tosses are not needed, so that we finish generating $d_i(x)$ and get $\sigma(x)$. Thus the calculation of $\sigma(x)$ ends in a finite time with probability 1.

A function $\tau : \mathbb{T}^1 \to \mathbb{N} \cup \{\infty\}$ is called a $\{\mathcal{B}_m\}_m$- *stopping time* (cf.[5, 7]), if $\{\tau \leq m\} := \{x \in \mathbb{T}^1 \mid \tau(x) \leq m\} \in \mathcal{B}_m$, $\forall m \in \mathbb{N}$. For a $\{\mathcal{B}_m\}_m$-stopping time τ, we define a sub σ-field $\mathcal{B}_\tau$ of $\mathcal{B}$ by $\mathcal{B}_\tau := \{A \in \mathcal{B} \mid A \cap \{\tau \leq m\} \in \mathcal{B}_m, \forall m \in \mathbb{N}\}$. A function $f : \mathbb{T}^1 \to \mathbb{R} \cup \{\pm\infty\}$ is $\mathcal{B}_\tau$-measurable, if and only if $f(x) = f\left(\lfloor x \rfloor_{\tau(x)}\right)$, $x \in \mathbb{T}^1$.

The random variable σ in Example 3.3 is a $\{\mathcal{B}_m\}_m$-stopping time, and σ itself is $\mathcal{B}_\sigma$-measurable. In general, if a $\{\mathcal{B}_m\}_m$-stopping time τ satisfies $\mathbb{P}(\tau < \infty) = 1$, any $\mathcal{B}_\tau$-measurable function f can be calculated with high probability. Such f is called *simulatable* (cf. [7]).

If a stopping time τ satisfies $\mathbf{E}[\tau] < \infty$, it is easy to apply i.i.d.-sampling to any $\mathcal{B}_\tau$-measurable f. Indeed, putting

$$\mathbf{z}_n(x) := \lfloor 2^{\sum_{i=1}^{n-1} \tau(\mathbf{z}_i(x))} x \rfloor_{\tau(\mathbf{z}_n(x))}, \quad x \in \mathbb{T}^1, \quad n \in \mathbb{N},$$

since τ is a stopping time with $\mathbb{P}(\tau < \infty) = 1$, each $\mathbf{z}_n$ is defined with probability 1. $\{f(\mathbf{z}_n)\}_{n=1}^\infty$ on the Lebesgue probability space is an i.i.d.-sequence whose common distribution is equal to that of f.

Furthermore, we can apply pairwise independent sampling to any simulatable random variables.

Definition 3.1 (*Dynamic random Weyl sampling*(DRWS), [27, 30]) Let $j, K \in \mathbb{N}$, let $\{(x_l, \alpha_l)\}_{l \in \mathbb{N}}$ be $\Sigma^{K+j} \times \Sigma^{K+j}$-valued uniform i.i.d. random variables, and let τ be a $\{\mathcal{B}_m\}_m$-stopping time such that $\mathbf{E}[\tau] < \infty$. Define $\mathbb{T}^1$-valued random variables $\{\mathbf{x}_n\}_{n=1}^{2^j}$ by

$$\begin{aligned} \mathbf{x}_n &:= \sum_{l=1}^{\lceil \tau(\mathbf{x}_n)/K \rceil} 2^{-(l-1)K} \lfloor x_l + \nu_{n,l}\alpha_l \pmod 1 \rfloor_K, \\ \nu_{n,l} &:= \#\{1 \leq i \leq n \mid \tau(\mathbf{x}_i) > (l-1)K\}. \end{aligned}$$

Note that since τ is a $\{\mathcal{B}_m\}_m$-stopping time with $\mathbb{P}(\tau < \infty) = 1$, each $\nu_{n,l}$ and $\mathbf{x}_n$ are well-defined with probability 1.

Theorem 3.2 ([27]) *If f is $\mathcal{B}_\tau$-measurable, each $f(\mathbf{x}_n)$, $1 \le n \le 2^j$, has the same distribution as f, and $\{f(\mathbf{x}_n)\}_{n=1}^{2^j}$ are pairwise independent.*

4 Pseudo-random generator

4.1 General setting

A function $g_n : \Sigma^n \to \Sigma^{\ell(n)}$ with a parameter $n \in \mathbb{N}$ is called a *pseudo-random generator*, if $n < \ell(n)$. Suppose that Alice picks up an $\omega_n \in \Sigma^n$ at random, i.e., ω is assumed to be a random element with distribution P_n, which is called the *seed.* The output $g_n(\omega_n) \in \Sigma^{\ell(n)}$ is called *pseudo-random bits*, or *pseudo-random numbers.* As $n < \ell(n)$, $g_n(\omega_n)$ is not a coin tossing process.

For $A \subset \Sigma^{\ell(n)}$ with $P_{\ell(n)}(A) \ll 1$, we define a test for pseudo-random bits $g_n(\omega_n)$ whose rejection region is A. That is, If $P_n(g_n(\omega_n) \in A)$ is close to the significant level $P_{\ell(n)}(A)$, we admit that the pseudo-random bits $g_n(\omega_n)$ are accepted.

Expanding the above idea a little, let us formulate the notion of 'random test' for the pseudo-random generator g_n. For a function $A_n : \Sigma^{\ell(n)} \times \Sigma^{s(n)} \to \{0,1\}$ with parameter $n \in \mathbb{N}$, we define

$$\delta(g_n, A_n) := \left| P_n \otimes P_{s(n)} \left(A_n(g_n(\omega_n), \omega_{s(n)}) = 1 \right) - P_{\ell(n)} \otimes P_{s(n)} \left(A_n(\omega_{\ell(n)}, \omega_{s(n)}) = 1 \right) \right|.$$

Here '$\otimes$' stands for the direct product of probability measures. The quantity $\delta(g_n, A_n)$ indicates how A_n can distinguish $g_n(\omega_n)$ from real coin tosses $\omega_{\ell(n)}$. We introduced a random element $\omega_{s(n)}$ with distribution $P_{s(n)}$ which is independent of ω_n so as to describe a random test.

4.2 Secure pseudo-random generator

4.2.1 Definition

Since $n < \ell(n)$, there exists an A_n such that $\delta(g_n, A_n) \ge 1/2$ (for example, define A_n as the indicator function of the range of g_n). Hence it does not make sense if we consider all tests, i.e., all functions A_n. We must consider some restricted classes of A_n. Then for each class, there corresponds a class of pseudo-random generators which are accepted by every test of that class.

Now, under what standard shall we define the classes? In cryptography, they set computational complexity as the standard. Namely, $T(A_n)$ being the time complexity of A_n, they define

$$S(g_n, A_n) := \frac{T(A_n)}{\delta(g_n, A_n)}.$$

Suppose $S(g_n, A_n)$ is very large for all A_n. This means that if $T(A_n)$ is not so large, $\delta(g_n, A_n)$ must be very small, and that if $T(A_n)$ is very large, $\delta(g_n, A_n)$ need be not so small. They consider such g_n to be excellent.

As a theoretical research object, the following class of pseudo-random generators is the simplest and most investigated one.

Definition 4.1 Assume that $\ell(n)$ and $T(g_n)$ are at most of polynomial growth in n. If $S(g_n, A_n)$ is of over polynomial growth for any A_n, we say g_n is a *secure* pseudo-random generator, or that the pseudo-random bits $g_n(\omega_n)$ is secure.

Remark 4.1 For example, $e^{t/1000}$ grows much faster than all polynomials as $t \to \infty$, but for small t, it is much smaller than t^{100}. Thus in practical scale, a function of polynomial time complexity is not necessarily feasible, and a function of over polynomial complexity is not necessarily infeasible. In practice, we should investigate whether feasible or not case by case.

4.2.2 Secure pseudo-random generator and Monte Carlo method

Secure pseudo-random generator originally means that when we used it for cryptography, we can keep secret of communications with very high probability. But at the same time, it is also 'secure' in Monte Carlo method in the sense that its output can hardly be distinguished from real coin tossing process. Let us explain this fact.

Let X be a random variable with $\ell(n)$-bit randomness. Suppose as in § 3.3 that $\ell(n)$ is too large, and that we use a pseudo-random generator $g_n : \Sigma^n \to \Sigma^{\ell(n)}$ for the input of X. Namely, we use $X' = X(g_n(\omega_n))$ instead of X in practical numerical calculus. Then the question is whether or not the distribution of X' is close to that of X.

To answer this question, let us compare their distribution functions $F_X(t) := P_{\ell(n)}(X \leq t)$ and $F_{X'}(t) := P_n(X' \leq t)$. If g_n is secure, then it is assured that $F_X(t)$ and $F_{X'}(t)$ are close to each other for any t, if n is large enough. Indeed, defining a function for test A_n by $A_n(\omega_{\ell(n)}) := \mathbf{1}_{\{X(\omega_{\ell(n)})\leq t\}}$, $\omega_{\ell(n)} \in \Sigma^{\ell(n)}$, that X is practically computable implies $T(A_n)$ is rather small. Then by the definition of secure pseudo-random generator, we see that

$$|F_X(t) - F_{X'}(t)| \;=\; \left|P_{\ell(n)}\left(A_n(\omega_{\ell(n)}) = 1\right) - P_n\left(A_n(g_n(\omega_n)) = 1\right)\right|$$

must be very small.

4.2.3 Existence problem

From theoretical point of view, the class of secure pseudo-random generators is the simplest and most natural class among all classes of pseudo-random generators. However, unfortunately, we do not know whether it is empty or not. This existence problem is deeply related to '$\mathbf{P} \neq \mathbf{NP}$ conjecture', which is one of the most important unsolved problems in theory of computational complexity. That is, if $\mathbf{P} = \mathbf{NP}$, there exists no secure pseudo-random generator$^{2)}$.

Nevertheless, researchers are optimistic. They think that of course it is excellent if it exists, but even if it does not, there would be some progress in $\mathbf{P} \neq \mathbf{NP}$ conjecture. Anyway, while the negative answer is not proved, let us assume its existence.

4.2.4 Next-bit-prediction

The notion of 'next-bit-prediction' is a clue to how we construct a secure pseudo-random generator.

Definition 4.2 Let $g_n : \Sigma^n \to \Sigma^{\ell(n)}$ be a pseudo-random generator and let $P_{\ell(n),n}$ be the uniform probability measure on the product set $\{1, 2, \ldots, \ell(n)\} \times \Sigma^n$. In what follows, $(I, \omega_n) \in \{1, 2, \ldots, \ell(n)\} \times \Sigma^n$ is an random element with distribution $P_{\ell(n),n}$. For each function $\tilde{A} : \{1, 2, \ldots, \ell(n)\} \times \Sigma^{\ell(n)} \times \Sigma^{s(n)} \to \{0, 1\}$, we define

$$\tilde{\delta}(g_n, \tilde{A}_n) \;:=\; P_{\ell(n),n} \otimes P_{s(n)} \Big(\tilde{A}_n\left(I, \lfloor g_n(\omega_n)\rfloor_{I-1}, \omega_{s(n)}\right) = d_I(g_n(\omega_n))\Big) - \frac{1}{2}.$$

This is the probability that $\tilde{A}_n$ exactly predicts the I-th bit $d_I(g_n(\omega_n))$ from know $I-1$-bits $\lfloor g_n(\omega_n)\rfloor_{I-1}$. Then if for any $\tilde{A}_n$,

$$\tilde{S}(g_n, \tilde{A}_n) := \left|\frac{T(\tilde{A}_n)}{\tilde{\delta}(g_n, \tilde{A}_n)}\right|$$

is of over polynomial growth, g_n is said to be *next-bit-unpredictable.*

Since next-bit-prediction can be regarded as a special kind of tests mentioned in § 4.2.1, a secure pseudo-random generator is of course next-bit-unpredictable. Interesting is that the converse also holds.

Theorem 4.1 (cf. [19, 24]) *A pseudo-random generator is secure if and only if it is next-bit-unpredictable.*

By Theorem 4.1, in order to construct a secure pseudo-random generator, it is enough to design one to be next-bit-unpredictable. Some general methods are known by which we can construct a next-bit-unpredictable pseudo-random generator, assuming the existence of one-way function (function itself is feasible but its inverse is not feasible), which assumption is stronger than $\mathbf{P} \neq \mathbf{NP}$. The BBS generator([3, 4, 24, 37]) is the first one that is considered to be a next-bit-unpredictable, which stands on the assumption that a product of two big prime numbers are very hard to factorize.

4.3 Pseudo-random generator for special use

Until now, we have been tacitly considering all-purpose pseudo-random generator. But in Monte Carlo method, pseudo-random generators are often used for special purposes. If we restrict the use, it is not surprising that pseudo-random bits with small randomness may have the same function as coin tossing process for that use, irrelevantly to $\mathbf{P} \neq \mathbf{NP}$ conjecture. It is because the restriction of the use means that the class of tests by which the pseudo-random bits should be accepted may be determined without any ground of computational complexity.

Such an example has already be seen in numerical integration. Indeed, (D)RWS has the same mean square error as i.i.d.-sampling. And the former requires much less randomness than the latter does. That is, the random variables generated by (D)RWS are not accepted by every test, but are accepted by the test which asks if they have the same function as i.i.d. random variables in numerical integration.

In this sense, (D)RWS is a pseudo-random generator exclusively adapted to numerical integration.

5 Security of pseudo-random generator and complexity

5.1 Disappearance of dependence

In Fig. 1 of § 3.1, we saw the error $|N^{-1}\sum_{n=1}^{N} f_{50}(n\alpha) - (1/2)|$ decreases at the rate of $O(N^{-1/2})$. We can say this is an average rate according to Theorem 3.1. In fact, it is widely known that when integrands are very complicated, quasi-Monte Carlo method seems to converge as slow as i.i.d.-sampling([7]). This phenomenon is intuitively explained as follows: It is essentially difficult to integrate very complicated functions numerically, so that any method does not efficiently work for them. Consequently, any method can give at most the same convergence rate as i.i.d.-sampling.

Now, isn't it natural to imagine that the samples $\{f(x_n)\}_n$ generated by quasi-Monte Carlo method applied to a very complicated function f look very random? Indeed, such 'disappearance of dependence' occurs in several examples, such as:

Theorem 5.1 ([10]) *Let $S_m(x)$ be the function defined by (1). Then for $\mathbb{P}$-a.e. α, the stochastic process $\{2m^{-1/2}(S_m(\,\cdot+n\alpha)-(m/2))\}_{n=1}^{\infty}$ on $(\mathbb{T}^1,\mathcal{B},\mathbb{P})$ converges in each finite dimensional distribution to i.i.d.-random variables with standard normal distribution as $m\to\infty$.*

Each 1-dimensional distribution of this process converges to standard normal distribution as $m\to\infty$ by the central limit theorem. It is not easy to see the process converges to a Gaussian system, but instead, let us see why the correlation vanishes.

First, we have $S_m(x)-(m/2)=(r_1(x)+\cdots+r_m(x))/2$, where $r_i(x):=1-2d_i(x)$ is the Rademacher functions. Let

$$\varphi(\alpha) := \frac{1}{4}I[r_1(\cdot)r_1(\cdot+\alpha)] = \left|\frac{1}{2}-\alpha\right|-\frac{1}{4}. \tag{19}$$

Noting that

$$r_i(x) = r_1(2^{i-1}x), \qquad I[r_i(\cdot)r_j(\cdot+\beta)] = 0, \quad \forall\beta, \quad i\neq j,$$

let us calculate the correlation:

$$\begin{aligned}
&I\left[m^{-1/2}\left(S_m(\cdot)-\frac{m}{2}\right)\cdot m^{-1/2}\left(S_m(\cdot+n\alpha)-\frac{m}{2}\right)\right]\\
&\quad= \frac{1}{4m}\sum_{i=1}^{m}\sum_{j=1}^{m}I[r_i(\cdot)r_j(\cdot+n\alpha)] = \frac{1}{4m}\sum_{i=1}^{m}I[r_i(\cdot)r_i(\cdot+n\alpha)]\\
&\quad= \frac{1}{4m}\sum_{i=1}^{m}I[r_1(2^{i-1}\cdot)r_1(2^{i-1}\cdot+2^{i-1}n\alpha)] = \frac{1}{4m}\sum_{i=1}^{m}I[r_1(\cdot)r_1(\cdot+2^{i-1}n\alpha)]\\
&\quad= \frac{1}{m}\sum_{i=1}^{m}\varphi(2^{i-1}n\alpha).
\end{aligned} \tag{20}$$

The last line converges to $I[\varphi]=0$ for a.e.α on account of the ergodicity of the transformation $\alpha\mapsto 2\alpha$ ([36]). But the convergence (20) is as slow as $O(m^{-1/2})$ ([10, 15]).

In the following theorem, much faster disappearance of dependence occurs.

Theorem 5.2 ([1, 25, 29, 34, 38, 39]) *Let $\{X_n^{(m)}(\cdot\,;\alpha)\}_{n=1}^{\infty}$, $\alpha\in\mathbb{T}^1$, be a $\{0,1\}$-valued stochastic process on $(\mathbb{T}^1,\mathcal{B},\mathbb{P})$ defined by*

$$X_n^{(m)}(x;\alpha) := S_m(x+n\alpha) \bmod 2, \quad x\in\mathbb{T}^1, \quad n=1,2,\ldots \tag{21}$$

Then for $\mathbb{P}$-a.e. α[3]*, $\{X_n^{(m)}(\cdot\,;\alpha)\}_{n=1}^{\infty}$ converges in each finite dimensional distribution to coin tossing process as $m\to\infty$. Furthermore, for $\mathbb{P}$-a.e. α, this convergence is exponentially fast in m.*

5.2 Connection to next-bit-prediction

It is interesting to look at Theorem 5.2 from the viewpoint of computational complexity. We can imagine that since the complexity of the function $S_m(x) \bmod 2$ increases as $m\to\infty$, any next-bit-prediction for $\{X_n^{(m)}(\cdot\,;\alpha)\}_{n=1}^{\infty}$ becomes more and more difficult. This must be reflected in Theorem 5.2.

Let us explain the situation a little more precise. First, we have an algorithm to calculate the following probability explicitly([25]).

$$F^{(m)}(k_0, k_1, \ldots, k_{l-1}; \alpha) := \mathbb{P}\left(\sum_{j=0}^{l-1} X_{k_j}^{(m)}(\,\cdot\,; \alpha) = \mathrm{odd}\right).$$

So we can consider following the next-bit-prediction:

$$\tilde{A}(x_1, x_2, \ldots, x_{k_{l-1}-1}) := \begin{cases} \mathbf{1}_{\{\sum_{j=0}^{l-2} x_{k_j} = \mathrm{even}\}}, & \text{if } F^{(m)}(k_0, k_1, \ldots, k_{l-1}; \alpha) > \frac{1}{2}, \\ \mathbf{1}_{\{\sum_{j=0}^{l-2} x_{k_j} = \mathrm{odd}\}}, & \text{if } F^{(m)}(k_0, k_1, \ldots, k_{l-1}; \alpha) < \frac{1}{2}. \end{cases}$$

It is easy to see

$$\mathbb{P}\left(\tilde{A}(X_1^{(m)}(\,\cdot\,; \alpha), \ldots, X_{k_{l-1}-1}^{(m)}(\,\cdot\,; \alpha)) = X_{k_{l-1}}^{(m)}(\,\cdot\,; \alpha)\right) = \left|F^{(m)}(k_0, k_1, \ldots, k_{l-1}; \alpha) - \frac{1}{2}\right| + \frac{1}{2}.$$

Namely, by $\tilde{A}$, we can predict the next bit with probability bigger than 1/2 by

$$\left|F^{(m)}(k_0, k_1, \ldots, k_{l-1}; \alpha) - \frac{1}{2}\right|.$$

But the last quantity converges for $\mathbb{P}$-a.e. α to 0 exponentially in m. This fact shows that the above next-bit-prediction becomes rapidly impossible in practice as m increases.

Although our next-bit-predictions are very limited ones, we think it very interesting to be able to see the connection between the complexty of $S_m(x) \bmod 2$ and next-bit-prediction via analysis in this way. Is it too optimistic to expect from the above fact that the process $\{X_n^{(m)}(\cdot\,; \alpha)\}_{n=1}^{\infty}$ defined by (21) is secure pseudo-random bits?

Incidentally, by Theorem 5.1, $\{f_m(\cdot + n\alpha)\}_{n=1}^{\infty}$ defined by (2) converges to coin tossing process for $\mathbb{P}$-a.e. α as $m \to \infty$, too, but the correlation decays as slow as $O(m^{-1/2})$, which implies $\{f_m(\cdot + n\alpha)\}_{n=1}^{\infty}$ is by no means secure as pseudo-random bits.

6 Conclusion

In this article, we used the term 'complicated function' for many times including the title. This does not simply mean 'function with large total variation'. Indeed, $d_{100}(x)$ has very large total variation, but the numerical integration by means of the Weyl sequence (4) is easy. It vaguely means 'function with large computational complexity', but we do not know clearly what notion we wish to express by the term. Anyway, what we wish to know is some quantitative relation between 'complexity' and analytic (in particular, probabilistic) phenomena. For example, consider, the two stochastic processes $\{f_{50}(\cdot + n\alpha)\}_{n=1}^{\infty}$ and $\{S_{50}(\cdot + n\alpha) \bmod 2\}_{n=1}^{\infty}$, both defined on $(\mathbb{T}^1, \mathcal{B}, \mathbb{P})$. As mentioned in § 5.2, the latter is much closer to coin tossing process than the former. How this phenomenon is quantitatively related to the complexity of f_{50} and $S_{50} \bmod 2$ attracts us very much.

Notes

1) The idea which the author thinks most reasonable. Some people may think in other ways.
2) **P** $\neq$ **NP** does not imply the existence of secure pseudo-random generator.
3) Recently, K.Ysutomi proved that for any irrational $\alpha \in \mathbb{T}^1$, the process $\{X_n^{(m)}(\cdot\,; \alpha)\}_{n=1}^{\infty}$ converges in law to coin tossing process as $m \to \infty$ ([40]).

References

[1] Y.Akine, Precise estimate for convergence rate of dependency vanishing theorem (in Japanese), Master thesis, Kyushu University, (2002).

[2] N.S. Bahvalov, Optimal convergence bounds for quadrature processes and integration methods of Monte Carlo type for classes of functions, (Russian), Ž. Vyčisl. Mat. i Mat. Fiz., **4-4** (1964), suppl., 5–63.

[3] L. Blum, M. Blum and M. Shub, A simple unpredictable pseudorandom number generator, SIAM J. Comput., **15-2** (1986), 364–383.

[4] M. Blum and S. Macali, How to generate cryptographically strong sequences of pseudo-random bits, SIAM J. on Computing, **13** (1984), 850–864. A preliminary version appears in Proceedings of the IEEE Foundations of Comput. Sci., (1982), 112–117.

[5] P. Billingsley, Probability and measure, 3rd edition, John Wiley & Sons, (1995).

[6] E. Borel, Sur les probabilités dénombrables et leurs applications arithmétiques, Circ. Mat. d. Palermo, **29** (1909), 247–271.

[7] N. Bouleau and D. Lépingle, Numerical methods for stochastic processes, John Wiley & Sons, (1994).

[8] G.J. Chaitin, Algorithmic information theory, IBM J. Res. Develop., **21** (1977), 350–359.

[9] B. Chor and O. Goldreich, On the power of two-point based sampling, J. Complexity, **5-1** (1989), 96–106.

[10] K. Fukuyama, The central limit theorem for Rademacher system, Proc. Japan Acad., **70**, Ser. A, No.7 (1994), 243–246.

[11] K. Fukuyama, Riesz-Raikov sums and Weyl transform, Monte Carlo Methods and Applications, VSP, **2-4** (1996), 271–293.

[12] O. GoldreichOded and A. Wigderson, Tiny families of functions with random properties: a quality-size trade-off for hashing, Proceedings of the Workshop on Randomized Algorithms and Computation (Berkeley, CA, 1995). Random Structures Algorithms, **11-4** (1997), 315–343.

[13] Procedure for random number generation and randomization (in Japanese), JIS Z 9031:2001, Japanese Standards Association (2001 revised).

[14] A. Joffe, On a set of almost deterministic k-independent random variables, Ann. Probability, **2-1** (1974), 161–162.

[15] M. Kac, On the distribution of values of sums of type $\sum f(2^k t)$, Ann. Math., **47** (1946), 33–49.

[16] A.N. Kolmogorov, Logical basis for information theory and probability theory, IEEE Trans. on Inform. Theo., vol.IT-**14-5**, Sept. (1968), 662–664.

[17] L. Kuipers and H. Niederreiter, Uniform distribution of sequences, Interscience, (1974).

[18] D.E. Knuth, The Art of Computer Programming, 2nd ed., Addison-Wesley, (1981).

[19] M. Luby, Pseudorandmness and cryptographic applications, Princeton Computer Science Notes, Princeton University Press, (1996).

[20] P. Martin-Löf, The definition of random sequences, Inform. Control, **7** (1966), 602–619.

[21] M. Matsumoto and T. Nishimura, Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator, ACM. Trans. Model. Comput. Simul., **8-1**, (1998) 3–30.

[22] H. Niederreiter, Quasi-Monte Carlo methods and pseudo-random numbers, Bull. Amer. Math. Soc., **84** (1978), 957–1041.

[23] J. von Neumann, Various techniques used in connection with random digits, U.S. Natl. Bur. Stand. Appl. Math. Ser., **12** (1951), 36–38.

[24] D.R. Stinson, Cryptography (Theory and practice), CRC Press, Boca Raton/ Ann Arbor / London / Washington,D.C., (1995).

[25] H. Sugita, Pseudo-random number generator by means of irrational rotation, Monte Carlo Methods and Applications, VSP, **1-1** (1995), 35–57.

[26] H. Sugita, Robust numerical integration and pairwise independent random variables, Jour. Comput. Appl. Math., **139** (2002), 1–8.

[27] H. Sugita, Dynamic random Weyl sampling for drastic reduction of randomness in Monte Carlo integration, Math. Comput. Simulation, **62** (2003), 529–537.

[28] H. Sugita, Monte-Carlo integration using cryptographically secure pseudo-random generator, Numerical Methods and Applications, Lecture Notes in Computer Science **2542**, Springer (2003), 140–146.

[29] H. Sugita, An analytic approach to secure pseudo-random generation, to appear in Proceedings of 2003 Ritsumeikan Symposium on Stochastic Processes and its Applications to Mathematical Finace, World Scientific, 2004.

[30] H.Sugita, *The Random sampler*, C and C++ libraries for pseudo-random generation and dynamic random Weyl sampling, available at `http://idisk.mac.com/hiroshi_sugita/Public/imath/mathematics.html`.

[31] H. Sugita and S. Takanobu, Limit theorem for symmetric statistics with respect to Weyl transformation: Disappearance of dependency, J. Math. Kyoto Univ., **38-4** (1998), 653–671.

[32] H. Sugita and S. Takanobu, Limit theorem for Weyl transformation in infinite-dimensional torus: Disappearance of dependency, J. Math. Sci. Univ. Tokyo, **7** (2000), 99–146.

[33] H. Sugita and S. Takanobu, Random Weyl sampling for robust numerical integration of complicated functions, Monte Carlo Methods and Appl., **6-1** (1999), 27–48.

[34] S. Takanobu, On the strong-mixing property of skew product of binary transformation on 2-dimensional torus by irrational rotation, Tokyo J. Math., **25-1** (2002), 1–15.

[35] S. Tezuka, Uniform Random Numbers: Theory and Practice, Kluwer International Series in Engineering and Computer Science, **315**, (1995).

[36] P. Walter, An introduction to ergodic theory, Springer, (1981).

[37] A. Yao, Theory and applications of trapdoor functions, Proceedings of the IEEE Foundations of Comput. Sci., (1982), 80–91.

[38] K. Yasutomi, A limit theorem for sequences generated by Weyl transformation: Disappearance of dependence, Probab. Theory Related Fields, **124-2** (2002), 178–188.

[39] K. Yasutomi, A direct proof of dependence vanishing theorem for sequences generated by Weyl transformation, Jour. Math. Kyoto Univ. **43-3** (2003), 599–607.

[40] K. Yasutomi, A dependence vanishing theorem for sequences generated by Weyl transformation, preprint.

(Hiroshi SUGITA, Graduate school of Mathematics, Osaka University)