Regular Paper

Multidimensional Uniformity of Pseudorandom and Quasirandom Sequences

Takao Tsuda[†]

It has been said that quasirandom sequences have a better uniform distribution in multidimensional spaces than pseudorandom sequences, and are therefore superior for numerical integration of multidimensional functions. In this paper, however, it is numerically demonstrated that there is a certain critical number of dimensions k_c between 20 and 40 dimensions, and that in higher dimensions than k_c, pseudorandom and Richtmyer sequences have lower discrepancy, and hence better uniformity, than quasirandom sequences, yielding substantially smaller errors to numerical integration.

1. Introduction

Monte Carlo methods with pseudorandom numbers have long been used for numerical integration of functons defined in a multidimensional space. See, for example, Ref. 1). The number of dimensions of the numerical integration has reached as high as a few hundred (or even more) depending on the current application areas. Consider, for example, the search for a potential minimum of a many-atom molecule configuration using the Lennard-Jones potential model[2)]. The search relies on a genetic algorithm[3)], so that the best solution known at one time was actually the second best, and was replaced later by a better globally minimal solution. To remedy the incompetence of the algorithm, another algorithm has been proposed to verify whether the minimum captured is a global minimum[4)]. To execute this algorithm, one has to perform numerical integration in a 300-dimensional space when the problem on hand is a 100-atom cluster. Another example of current interest is the numerical solution of stochastic differential equations. The numerical computation can be recast as the numerical integration of a multiple integral[5)] whose number of dimensions amounts to the number of divisions of the time interval. If one divides the time interval over which the solution is sought into 100 equal subdivisions, then the number of dimensions of the integral is 100.

On the other hand, in a few monographs, it is conclusively stated that quasirandom sequences have better uniformity than pseudorandom sequences, and hence are superior to pseudorandom sequences for the purpose of numerical integration. See p.10 of Ref. 7) and p.182 of Ref. 6). The purpose of this paper is to demonstrate numerically that there is a certain critical number of dimensions, k_c, above which the uniformity is worse with quasirandom sequences than with pseudorandom sequences, and thus that the supposed superiority of quasirandom sequences with regard to numerical integration is questionable.

2. Errors of Numerical Integration and Discrepancy

2.1 A Point Sequence in a Multidimensional Space and the Definition of Discrepancy

Let the point sequence

$$P = \{\mathbf{x}_n, n = 0, 1, 2, \cdots, N-1\}$$

in a k-dimensional hypercube $[0,1]^k$, and let $\mathbf{y}$ $(=(y_1, y_2, \cdots, y_k))$ also belong to the same hypercube $[0,1]^k$. $E(\mathbf{y})$ is the partial volume $[0, y_1) \times [0, y_2) \times \cdots \times [0, y_k)$ in the k-dimensional hypercube. $\#(E(\mathbf{y}); N)$ denotes the number of points that have fallen inside the partial volume $E(\mathbf{y})$ among all the points $\mathbf{x}_n$ $(n = 0, 1, 2, \cdots, N-1)$ of the point sequence P. The discrepancy $D_N^{(k)}$ of the N-point set P in a k-dimensional space is defined to be

$$D_N^{(k)} = \sup_{\mathbf{y} \in [0,1]^k} \left| \frac{\#(E(\mathbf{y}); N)}{N} - \prod_{i=1}^{k} y_i \right| \quad (1)$$

Discrepancy is defined for any arbitrary and reproducible point sequence. It can therefore be defined for a pseudorandom sequence generated by arithmetic operations. The concept of discrepancy dates back to 1960 or earlier, although

† Faculty of Information Sciences, Hiroshima City University

the term itself had not been coined at that time, and can be found in Ref. 10), where not only the L_∞-norm as in the above Eq. (1) but also the L_2-norm was used. In this paper, all the objective sequences are predetermined sequences that include pseudorandom sequences, and therefore the definition of Eq. (1) by the L_∞ norm is considered.

2.2 Errors of numerical integration

By using the formula for discrepancy (1), the error of numerical integration is given by

$$\left| \int_{[0,1]^k} f(\mathbf{x})d\mathbf{x} - \frac{1}{N}\sum_{i=0}^{N-1} f(\mathbf{x}_i) \right| \le V(f) D_N^{(k)} \tag{2}$$

where $V(f)$ is the *variation* of the function $f(\mathbf{x})$. For details, see p.181 of Ref. 6). $V(f)$ in the one-dimensonal case may be expressed as $\int_0^1 |df(x)|$. This differs from the probabilistic error bound of the Monte Carlo integration. It gives the deterministic error bound of numerical integration. In light of the definition $D_N^{(k)}$ the error bound can be used for any predetermined and reproducible sequences including pseudorandom sequences. The main issue is the behaviour of the discrepancy for very large values of k.

3. Main Random Number Generators

There are so many variants and details are not shown; readers are referred to Ref. 6) for more details. Here, only the main issues are summarized.

3.1 Pseudorandom Numbers (PRN for short)

3.1.1 Linear Congruential Methods

As is well known, pseudorandom numbers in this category have lattice structures whose nature has been well studied. These unfavourable structures can be removed by adding *shuffles.* Statistical tests corroborate the substantial improvement achieved by the use of shuffles [8]. The deficiency of random numbers of this type is that the period is too short for some applications. This defect can be averted by using 64-bit integer arithmetic. There has been some recent work to lengthen the period. Numerical observation by PRN is performed in this paper, using linear congruential pseudorandom numbers with shuffling added. **The shuffling algorithm** is as follows:

Those random numbers generated by the linear congruential method are stored temporarily in a number of entries of a table, and which one of them is to be picked out and used next is determined by the raw random number before shuffling. The table $T(i)$ $(i = 1, 2, \cdots, t)$ is provided for temporary storage. The number of entries, namely, the size of the table, should not be a power of 2. In the following, m is the largest integer plus 1, and hence is 2^{32} in the case of single-precision integers.

```
(1) $T(i) \leftarrow y_i$ $(i = 1, 2, \cdots, t)$
        [generate $t$ random numbers]
(2) $y \leftarrow y_{i+1}$
        [generate one more and let it be $y$ ]
(3) $j \leftarrow \lfloor ty/m \rfloor + 1$
        [pick out an entry by $y$ ]
(4) $y \leftarrow T(j)$; output $y/m$
        [output the random number normalized to [0,1]]
(5) $T(j) \leftarrow$ next $y$; return to (3) with this $y$
        [generate a new $y$ and repeat]
```

$\lfloor x \rfloor$ is the *floor* of x that gives an integer by ignoring those digits below the decimal point. The larger the table size is, the more the quality of the random numbers is improved. The number of entries should be greater than 15 [8].

Let pseudorandom numbers be generated by the linear congruence:

$$y_{i+1} = 11 \times y_i + 0 \pmod{2^{32}} \tag{3}$$

From the pseudorandom sequences thus generated, $y_0, y_1, y_2, \cdots$, let the x- and y-coordinates of a point be y_i/m and y_{i+1}/m, respectively. The distribution of 10,000 such points is shown in **Fig. 1**. The initial value is given by $y_0 = 100$.

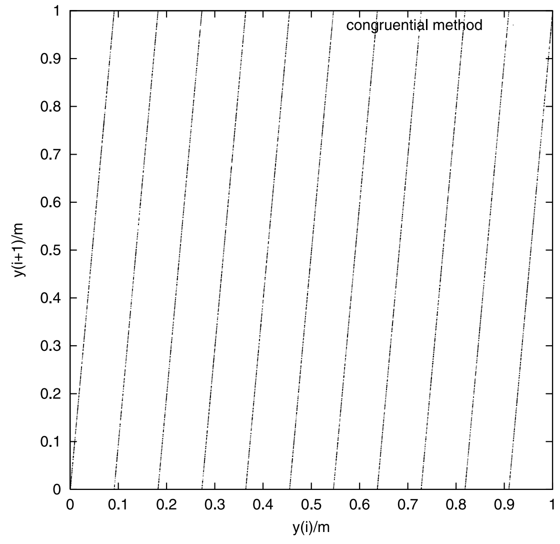

Fig. 1 Without shuffling.

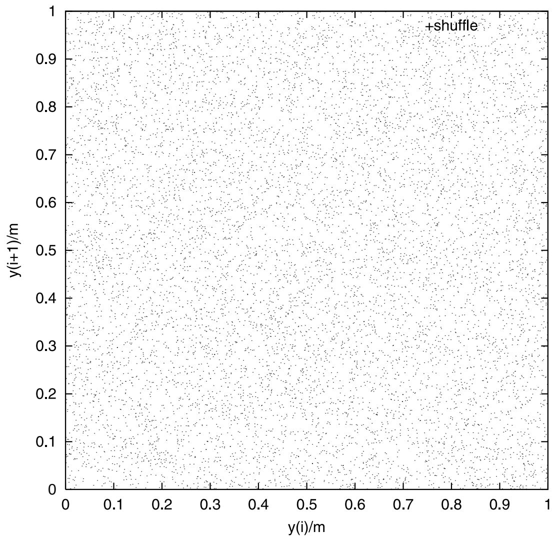

Fig. 2 With shuffling added.

Figure 2 shows the results obtained by using the same formula, the same initial value as Fig. 1, and a table size for shuffling of 100. The effect of shuffles is evident. (This can indeed be confirmed by statistical χ^2-tests for spatial uniformity.)

3.1.2 M Sequences (Generalized Feedback Shift Register: GFSR)

This type of random number generators also yield numbers with lattice structures, and is analyzed in detail[6]. There are many variants. The period can be extremely long, and it is impossible to exhaust the whole period by continual number generation on a computer. For applications, therefore, one actually uses only a short segment of the whole sequence. It is almost negligibly short as compared with the length of the whole period. One main defect of this type of generators is that there is no theoretical guarantee of the probabilistic quality over this *partial* sequence used in practice.

One may summarize as follows:
Pseudorandom numbers are *random* and *chaotic*. The term *random* means that numbers generated assume values that are amenable to probabilistic interpretation. The term *chaotic* means that the arrangement of point positions of the point sequence is in disorder, but yet it is such that the points cover the whole interval (or domain) of interest. The pseudorandom numbers are completely reproducible, and thus in this sense they are predetermined sequences for which the concept of discrepancy is applicable and the integration error is given by Eq. (2).

3.2 Quasirandom Numbers (QRN for short)

3.2.1 Richtmyer Sequence

QRN was first considered by Richtmyer[9),10)]. Although the term QRN was then used by Richtmyer, the Richtmyer sequence is not considered within the framework of today's QRN. As is shown later, however, the Richtmyer sequence shows good uniform distribution in both low and high multidimensional spaces. Take an algebraic irrational number (which may be called a *seed*) and multiply it by positive integers ($\times 1, \times 2, \times 3, \cdots$). The Richtmyer sequence uses the fractional parts of the numbers thus obtained. The time cost for generation is quite low; however, when a long sequence is needed, care must be exercised to compensate for the loss of lower significant digits as the multiplication proceeds. Let the Richtmyer sequence be called Richtmyer for short. As is described later, prime numbers are generated in ascending order. The first k of them are used as seeds, after square-root operations on each of them, for the generation of a k-dimensional point.

Richtmyer is not *random* but *chaotic*. On the same computer, it is a reproducible, and hence predetermined, sequence. The concept of discrepancy is again applicable, and the integration error is given by Eq. (2). The term *predictable* sequence is ambiguous, but at any rate Richtmyer can generate the i-th number of the sequence, no matter what i (> 0) is mentioned.

3.2.2 Halton and Other QRNs

Today, as QRN, such sequences as Hammersley, Halton, Faure, Sobol sequences and the generalized Niederreiter sequence (inclusive of those just mentioned) are known. See Ref. 6) for details. In this paper, the Halton sequence is taken as representative of all QRNs. Let the sequence be called Halton for short. The common feature of QRNs is that their theoretical upper bound of discrepancy is given by the same expression (4) below.

Unlike PRN (*random*) and Richtmyer (*chaotic*), QRNs as represented by Halton are such that the placement of the subsequent points of the point sequence is *regular*, and hence reproducible.

Let N points of $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \cdots$ be taken in the k-dimensional hypercube $[0,1]^k$. When the discrepancy $D_N^{(k)}$ of those points is bounded from above by Eq. (4), they are called a low-discrepancy sequence, or QRN. $c(k)$ of Eq. (4) depends only on the number of dimensions k.

$$D_N^{(k)} \le c(k)\frac{(\log N)^k}{N} \tag{4}$$

Halton, which is typical of QRNs, is subject to the following expression:

$$D_N^{(k)} \le c(b_1, \cdots, b_k)\frac{(\log N)^k}{N} + O\left(\frac{(\log N)^{k-1}}{N}\right) \tag{5}$$

where $c(b_1, \cdots, b_k) \approx \prod_{i=1}^{k} \frac{b_i}{\log b_i}$, so that for a large value of prime b used as a base, namely, in the case of high dimensionality, the value of $c(b_1, \cdots, b_k)$ may also be large, giving a large value to the discrepancy $D_N^{(k)}$.

Both Eqs. (4) and (5) give upper bounds of what is shown on the left side. This means that even if the right-hand side is large, the left-hand side, the discrepancy, may not be large. Concerning the lower bound of QRN's discrepancy, we have the conjecture:

$$D_N^{(k)} > c_1(k)\frac{(\log N)^{k-1}}{N} \tag{6}$$

which is supposed to hold for an arbitrary k dimensions ($k > 2$) and for an arbitrary point set in hypercube $[0,1]^k$, and where coefficient $c_1(k)$ is some positive constant that depends only on k. As will be demonstrated later in this paper, because the associated discrepancy $D_N^{(k)}$ does not increase with the increase of the number of dimensions k, Richtmyer and PRN do not comply with this conjecture.

4. Generation of a *k*-Dimensional Point Sequence

In the k-dimensoinal hypercube ($k = 1, 2, \cdots$), N points are generated one after another in order to observe the discrepancy. The method of point generation is explained below. While the number of k-dimensional points, N, is kept at 10^7 throughout the experiment, we vary the number of dimensions k up to 200.

4.1 *k*-Dimensional PRN

Given the initial value $y_0 = 123456$, the linear congruential method

$$y_{i+1} = 65531 \times y_i + 0 \pmod{m}, \quad m = 2^{32} \tag{7}$$

is dovetailed with the shuffling algorithm. Let the i-th random number thus obtained, and then normalized to a number over [0,1]-interval, be denoted by $prn(i)$. The table size (the number of entries) is 100. Lattice structures disappear as a result of shuffling. The coordinates of each point are shown in **Table 1**.

Table 1 PRN sequence ($0 \le k \le k_{\max}$).

No. of points	k-dimensional coordinates
point 1	$(prn(1), prn(2), prn(3), \cdots, prn(k))$
point 2	$(prn(k_{\max}+1), \cdots, prn(k_{\max}+k))$
point 3	$(prn(2k_{\max}+1), \cdots, prn(2k_{\max}+k))$
$\cdots$	$\cdots$
point N	$(prn((N-1)k_{\max}+1), \cdots, \cdots)$

Table 2 Richtmyer sequence.

No. of points	k-dimensional coordinates
point 1	$([1 \times b_1], [1 \times b_2], \cdots, [1 \times b_k])$
point 2	$([2 \times b_1], [2 \times b_2], \cdots, [2 \times b_k])$
point 3	$([3 \times b_1], [3 \times b_2], \cdots, [3 \times b_k])$
$\cdots$	$\cdots$
point N	$([N \times b_1], [N \times b_2], \cdots, [N \times b_k])$

4.2 *k*-Dimensional Richtmyer

k-dimensional Richtmyer points are generated, with k mutually independent algebraic irrational numbers used as seed numbers. In the case where the number of dimensions is k, let the i-th prime ($0 < i \le k$) be p_i. By taking the square root of each p_i, we have seed b_i ($= \sqrt{p_i}$), which will be multiplied by $1, 2, 3, \cdots$, and only the fractional parts are used. Since there are primes $2, 3, 5, 7, \cdots$, the first k-dimensional Richtmyer point is given by

$$(0.414213562\cdots, 0.732050808\cdots, 0.236067977\cdots, \cdots, [b_i], \cdots, [b_k]),$$

where square brackets $[x]$ indicate the fractional part of x. Thus the coordinates of the N k-dimensional Richtmyer points are as shown in **Table 2**.

Because of the finite word-length of a computer, the lower digits of an irrational number are lost when N is large. For a large value of N, one has to use an appropriate multiprecision integer arithmetic to guarantee correctness.

In both cases of PRN and Richtmyer, plots of points in partial 2-dimensions where the 99-th coordinate values are taken on the x-axis, and the 100-th coordinate values on the y-axis do not differ in uniformity from those plots where the first coordinate values are taken on the x-axis and the second coordinate values on the y-axis.

4.3 *k*-Dimensional Halton

Out of the QRNs known as Hammersley,

Table 3 Halton sequence.

No. of points	Coordinates (base 2, base 3, base 5)
point 1	$(0.5, 0.33\cdots, 0.2)$
point 2	$(0.25, 0.66\cdots, 0.4)$
point 3	$(0.75, 0.11\cdots, 0.6)$
$\cdots$	$\cdots$

Halton, Faure, Sobol, and generalized Niederreiter sequences, Halton is chosen as representative for the object of observation in this paper. There may be some difference in behaviour between these QRNs, but all of them share the same characteristic with regard to the discrepancy, which increases with the increase in the number of dimensions k.

Halton's point sequence is generated as follows. First, consider 1-dimensional Halton. With base b, which may be taken for prime 2 for example, integer n is transformed into a b-ary digit number:

$$a_m \times b^m + a_{m-1} \times b^{m-1} + \cdots + a_1 \times b + a_0,$$

where $0 \le a_j < b$, $0 \le j \le m$, $m = [\log n]$. The n-th number of Halton with base b, $\phi_b(n)$, is therefore given by

$$\phi_b(n) = \frac{a_0}{b} + \frac{a_1}{b^2} + \cdots + \frac{a_m}{b^{m+1}}. \tag{8}$$

In the case of 3 dimensions, for example, if the primes 2, 3, 5 are taken for the bases, the coordinates of the Halton points are as shown in **Table 3**.

In this way the coordinate values of the k-dimensional Halton points use, for bases b_i $(i = 1, 2, \cdots, k)$, the primes that are generated in ascending order with the initial prime being 2. Inevitably, the base value for a very high dimension becomes very large, and the time cost for generating the corresponding coordinate value substantially increases.

In the case of Halton, points plotted in partial 2 dimensions, where the 99-th coordinate values ($b_{99} = 523$) are taken on the x-axis and the 100-th coodinate values ($b_{100} = 541$) on the y-axis, are remarkably bad in uniformity as compared with those plots where the first coordinate values ($b_1 = 2$) are taken on the x-axis and the second coordinate values ($b_2 = 3$) on the y-axis. **Figures 3** and **4** show the point distribution, where the number of points in each point sequence is 10,000.

5. Observation of Discrepancy

5.1 Method of Observation and the Significance

Discrepancy (1) based on the L_∞-norm is defined as the supremum of the absolute values of the algebraic difference between the observed frequencies of k-dimensional points falling in the infinite number of hyperrectangles $[0, y_1) \times [0, y_2) \times \cdots \times [0, y_k)$ (where $\mathbf{y} \in [0,1]^k$) and the respective theoretical frequencies (proportional to the volumes of the hyperrectangles). The *test* hyperrectangles, so to say, are infinite in number, and what is more, the supremum is required, so that the theoretical value of the discrepancy is difficult to obtain by numerical measurement. By way of approximation, there-

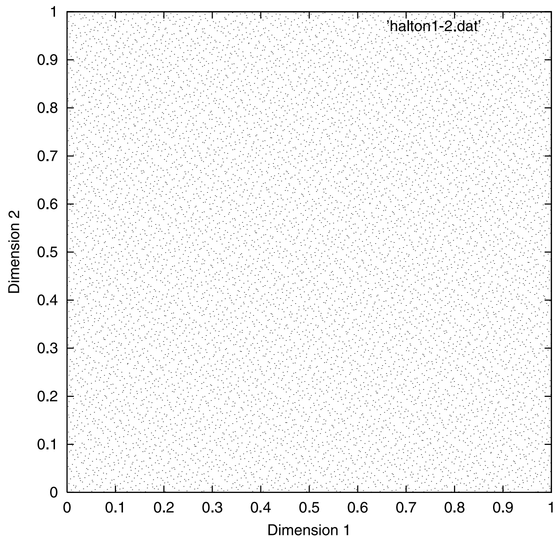

Fig. 3 Halton dim. 1 and dim. 2.

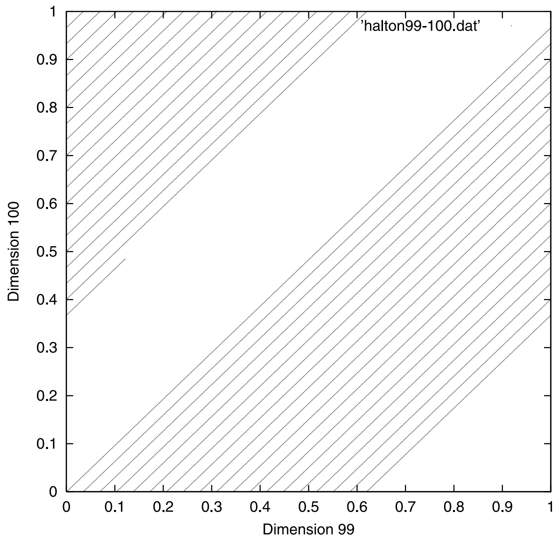

Fig. 4 Halton dim. 99 and dim. 100.

fore, the infinite number of hyperrectangles are replaced by as many test hyperrectangles as possible, each having a different shape, and the observed maximum is taken for an approximate estimate of the supremum value. The number of points in the k-dimensional hypercube, N, is taken as large as the computer time allows, and in the subsequent observations N is kept at $N = 10^7$ throughout this paper. As already mentioned, the dependence of the discrepancy $D_N^{(k)}$ on the number of dimensions k is observed by extensively varying k. The upper limit of k in the observation is denoted by $k_{\max}$.

As regards the programming technique, the outermost loop generates a $k_{\max}$-dimensional point whose first k coordinates give a k-dimensional point ($1 \le k \le k_{\max}$). This is repeated N times, so that in the end a total of $N \times k_{\max}$ random numbers ($\in [0, 1]$) are generated and used. On each generation of a $k_{\max}$-dimensional point, inner loops do checking to see if the resulting k-dimensional point ($1 \le k \le k_{\max}$) has fallen inside the numerous k-dimensional test hyperrectangles that have been prepared as explained in Section 5.2.

In other words, the point sequence is fixed as generated, whereas the number of dimensions k and the size and shape of the target test hyperrectangle are varied. For each k ($1 \le k \le k_{\max}$) separately, the maximum of the absolute values of the difference between the observed frequency and the corresponding theoretical frequency is recorded.

The relation between the observed discrepancy, $D_N^{(k)}(\mathit{observed})$, the true discrepancy (1), $D_N^{(k)}$, and its theoretical upper bound (4) is:

$$\begin{aligned} D_N^{(k)}(\mathit{observed}) &< D_N^{(k)}(\mathit{true}) \\ &\le D_N^{(k)}(\mathit{upperbound}) \qquad (9) \end{aligned}$$

This means that the true value of the discrepancy is always worse (or larger) than the observed value. Two kinds of observations were carried out: one for the case where $k_{\max} = 100$, called **Observation 1**, and the other for the case where $k_{\max} = 200$, called **Observation 2**. The total number of test hyperrectangles of Observation 2 was reduced to one fourth of that in Observation 1, since otherwise the computation time would have become prohibitive.

Suppose the length of a peripheral edge is 0.5. A 100-dimensional hypercube would then have a volume of 2^{-100}, and this volume would be none but zero when compared with 1, the unit hypercube volume, in floating-point arithmetic. This suggests that the selection of test hyperrectangles is not easy but really counts.

5.2 Results of Observation

In place of infinitely many hyperrectangles $[0, y_1) \times [0, y_2) \times \cdots \times [0, y_k)$ for $\forall \mathbf{y} \in [0, 1]^k$, numerous hyperrectangles are considered for the purpose of observation and those points which fall inside them are counted. Each of the following $\mathit{regionX}$ ($X = 1, \cdots, 8$) is defined as an array with 10 elements:

- Region of the first observation:
$$\begin{aligned} \mathit{region}1 = \{&0.1, 0.2, 0.3, 0.4, 0.5, 0.6, \\ &0.7, 0.8, 0.9, 0.99\} \end{aligned}$$
- Region of the second observation:
$$\begin{aligned} \mathit{region}2 = \{&0.9, 0.95, 0.3, 0.8, 0.85, \\ &0.75, 0.2, 0.9999, 0.1, 0.5\} \end{aligned}$$
- Region of the third observation:
$$\begin{aligned} \mathit{region}3 = \{&0.95, 0.99, 0.93, 0.5, 0.97, \\ &0.91, 0.999, 0.7, 0.995, 0.9\} \end{aligned}$$
- Region of the fourth observation:
$$\begin{aligned} \mathit{region}4 = \{&0.01, 0.05, 0.1, 0.15, 0.2, \\ &0.25, 0.3, 0.35, 0.4, 0.45\} \end{aligned}$$
- Region of the fifth observation:
$$\begin{aligned} \mathit{region}5 = \{&0.91, 0.92, 0.93, 0.94, 0.95, \\ &0.96, 0.97, 0.98, 0.99, 0.999\} \end{aligned}$$
- Region of the sixth observation:
$$\begin{aligned} \mathit{region}6 = \{&0.81, 0.82, 0.83, 0.84, 0.85, \\ &0.86, 0.87, 0.88, 0.89, 0.9\} \end{aligned}$$
- Region of the seventh observation:
$$\begin{aligned} \mathit{region}7 = \{&0.991, 0.992, 0.993, 0.994, \\ &0.995, 0.996, 0.997, 0.998, \\ &0.999, 0.9999\} \end{aligned}$$
- Region of the eighth observation:
$$\begin{aligned} \mathit{region}8 = \{&0.9999, 0.999, 0.998, 0.997, \\ &0.996, 0.995, 0.994, 0.993, \\ &0.992, 0.991\} \end{aligned}$$

As already stated, the total number of points in the point sequence is fixed at $N = 10^7$, and $N \times k_{\max}$ random numbers are used.

5.2.1 Observation 1 ($k_{\max} = 100$)

All of the above $\mathit{regionX}$ ($X = 1, \cdots, 8$) are used. Start with $\mathit{region}1$. First, in the outermost loop of the program, one $k_{\max}$-dimensional point is generated. The i-th coordinate value of this point is compared with the $i(\mathrm{mod}10)$-th element of $\mathit{region}1$ for

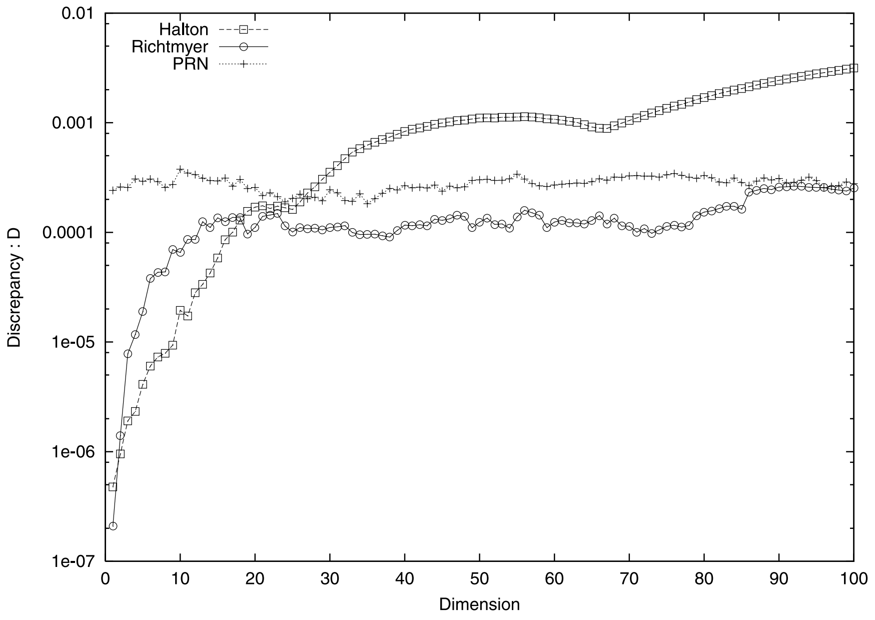

Fig. 5 Discrepancy up to 100 dimensions.

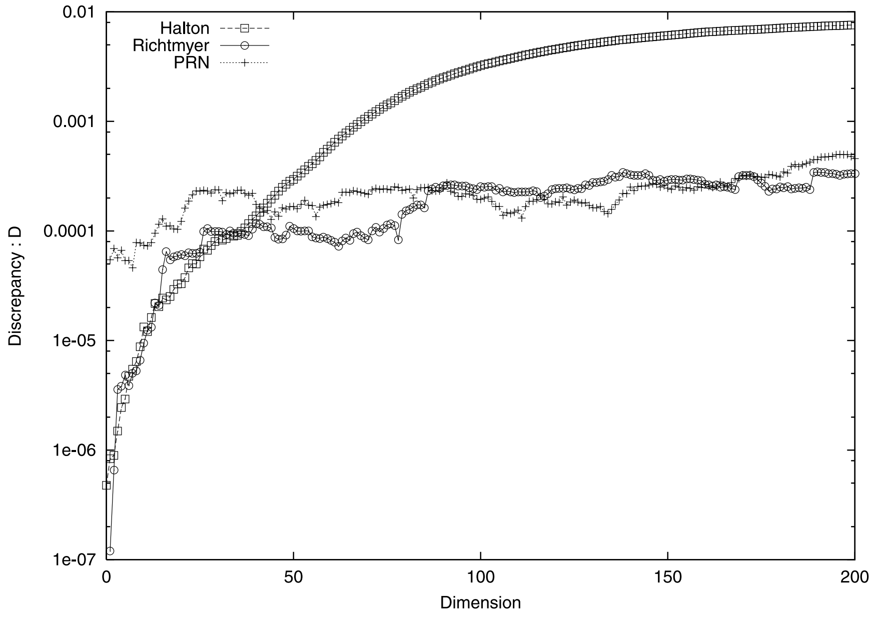

Fig. 6 Discrepancy up to 200 dimensions.

$i = 1, 2, \cdots, k$. If the i-th coordinate value is smaller than the corresponding element of *region*1 for all i, then this point is judged to have fallen inside the relevant k-dimensional test hyperrectangle, and is counted as such. This operation is repeated until $k = k_{\max}$. Next, each of the elements of *region*1 is cyclic-shifted in such a way as to yield {0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,0.99,0.1}. The same operation is repeated for these array elements. This cyclic shift is repeated as many times as the size of array *region*1 (namely, 10 times). Next, consider a k-dimensional hypercube whose edge length is given by the i-th element of *region*1 ($i = 1, 2, \cdots, 10$), and counting is done to see how many of the points of the point sequence have fallen inside the hypercube. For *region*1, there are 10 test hypercubes for each of k up to $k_{\max}$-dimensions. The whole process is repeated, by the outermost loop of the program, on the second point, the third point, $\cdots$ until all of the N points of the point sequence in $k_{\max}$-dimensions are exhausted. Since the point sequence remains the same, this means that the shapes of the test hyperrectangles have been changed. By means of these counting processes, comparisons are made against theoretical frequencies (namely, the volumes of test hyperrectangles) and the maximum of the absolute values of the differences is recorded.

After the same comparing and counting operations have been repeated over $region2, \cdots, region8$, the final maximum of the absolute values of the differences is obtained as $D_N^{(k)}(observed)$. The result of Observation 1 is shown in **Fig. 5**.

5.2.2 Observation 2 ($k_{\max} = 200$)

The range of the numbers of dimensions k is doubled. Because of the computing time, only $region5$ and $region6$ are used. Besides the computing time, the reason why these two *regions* are used is that, with increasing numbers of dimensions, the volumes of test hyperrectangles become much too small to allow the points falling therein to be counted, and most probably one has virtually $\frac{\#(E(\mathbf{y});N)}{N} = 0$; this situation is better avoided. The result of Observation 2 is shown in **Fig. 6**.

6. Conclusions

Discrepancy was defined as an index that designates the uniformity of the spatial distribution of multidimensional point sequences. The respective discrepancies of pseudorandom sequences, Richtmyer sequences, and quasirandom sequences were estimated by numerical observation. There are quite a few variants of quasirandom sequences, but they share the same characteristics with respect to the dimensional dependence. Halton sequences were chosen as representative of quasirandom numbers. The findings were that there is a certain critical number of dimensions, k_c, between 20 and 40 dimensions, and that in multidimensions higher than k_c, pseudorandom and Richtmyer sequences have smaller values of discrepancy than quasirandom sequences, and are thus superior to them for numerical integration. In multidimensions lower than k_c, Halton and Richtmyer sequences are, in that order, superior with regard to the values of discrepancy. On the whole, though Richtmyer sequences are not considered to be quasirandom sequences, they show good behaviour over the whole range of dimensions. Pseudorandom sequences, as dovetailed with shuffles, show good uniformity similar to Richtmyer sequences in multidimensions higher than k_c. One can therefore conclude that, although quasirandom sequences may be useful for numerical integration in low multidimensions, they are not necessarily superior in numerical integration in high multidimensions.

Acknowledgments As part of Bachelor's thesis, Masashi Matsumoto wrote programs with admirable skill, and obtained the numerical results described in this paper. The author is his supervisor, and expresses his appreciation to Matsumoto's painstaking effort.

References

1) Tsuda, T.: *Monte Carlo Methods and Simulation* (in Japanese), 3rd Edition, Baifukan Publishing, Tokyo (1995).

2) Northby, J.A.: Structure and Binding of Lennard-Jones Clusters: $13 < N < 147$, *J. Chem. Phys.*, Vol.87, No.10, pp.6166–6177 (1987).

3) Deaven, D.M. and Ho, K.M.: Molecular Geometry Optimization with a Genetic Algorithm, *Phys. Rev. Letters*, Vol.75, No.2, pp.288–291 (1995).

4) Tsuda, T. and Shibata, H.: Computational Verification on the Existence of the Global Maximum of a Multivariable Function (in Japanese), *Trans. IPS Japan*, Vol.21, No.6, pp.475–481 (1980).

5) Morokoff, W.J.: Generating Quasi-random Paths for Stochastic Processes, *SIAM Rev.*, Vol.40, No.4, pp.765–788 (1998).

6) Tezuka, S.: *Uniform Random Numbers: Theory and Practice*, Kluwer Academic Publishers (1995).

7) Niederreiter, H.: *Random Number Generation and Quasi-Monte Carlo Methods*, SIAM, Philadelphia (1992).

8) Tsuda, T.: Wrong Implementation of the Random Number Generator RANU 3 and Its Remedy (in Japanese), Kyoto University Research Institute for Mathematical Sciences (RIMS) Report 498 (Random Number Program Package), pp.49–56 (1983).

9) Richtmyer, R.D.: A Non-random Sampling Method, Based on Congruences, for Monte Carlo Problems, Inst. Math. Sci. New York Univ. Report NYO-8674 Physics (1958).

10) Hammersley, J.M. and Handscomb, D.C.: *Monte Carlo Methods*, Methuen & Co., London (1964); Hammersley, J.M.: Monte Carlo

Methods for Solving Multivariable Problems, *Ann. New York Acad. Sci.*, Vol.86, pp.844–874 (1960).

(Received April 5, 2000)
(Accepted September 7, 2000)

Takao Tsuda has been Professor Emeritus of Kyoto University, Information Science, since 1996. Currently with the Faculty of Information Sciences, Hiroshima City University as Professor of Computer Engineering. His field of interests is compiler techniques, such as vectorizing/parallelizing compilers, Monte Carlo methods, and magnetic reconnection. He is the author of *Monte Carlo Methods and Simulation* (in Japanese), 3rd Edition (Baifukan, 1995) and *Programming for Numerical Processing* (in Japanese), Iwanami Software Science Series, Vol.9 (Iwanami, 1988).