UCLA
COMPUTATIONAL AND APPLIED MATHEMATICS

Improved Stochastic Simulation Using Quasi-Monte Carlo: A Computational Study

Bradley Moskowitz

December 1994

CAM Report 94-40

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90024-1555

Improved Stochastic Simulation Using Quasi-Monte Carlo: A Computational Study

Bradley Moskowitz *
Mathematics Department, UCLA

August 10, 1994

Keywords: Monte Carlo, quasirandom sequence, stochastic process, simulation

Abstract

New methods for applying quasi-Monte Carlo to stochastic simulation are introduced. These include repeated renumbering, or scrambling, of simulation paths in order to enable low dimensional quasirandom points to be used without correlation errors. In the absence of renumbering, new dimensions are needed for each time step when using quasi-Monte Carlo, which is problematic because quasi-Monte Carlo loses its effectiveness with high dimensions. Encouraging computational results indicate that renumbering overcomes this difficulty and leads to more accurate results than either standard pseudorandom Monte Carlo or high dimensional quasi-Monte Carlo in the examples studied. The examples include a two dimensional randomized Lotka-Volterra system, a three dimensional Ornstein-Uhlenbeck process, and the hydrogen atom (using the Diffusion Monte Carlo method). In addition, two kinds of renumbering are compared: 'random' and 'binary' and it is found that binary renumbering leads to variation reduction and therefore reduced errors, which more than make up for the additional effort required.

*Los Angeles, California 90024. e-mail: mosko@math.ucla.edu. Research supported in part by the Air Force Office of Scientific Research under grant number F49620-94-1-0091.

1 Introduction

In attempting to apply quasi-Monte Carlo [1, 2, 3], which is simply the Monte Carlo method with quasirandom points used in place of standard pseudorandom points, to stochastic simulations, a typical difficulty is the high number of dimensions required [2, 4]. This is a consequence of the fact that in order to avoid correlation errors, new dimensions are needed for each time step of a simulation. As a result dimensions often become very high, and quasi-Monte Carlo is then no longer effective [1, 4, 5, 6, 7]. It is important to note that this problem is absent from standard pseudorandom Monte Carlo, and it is a consequence of the high degree of correlation between successive quasirandom points which enables them to achieve fast convergence rates but also introduces new difficulties such as the present one.

We have found that by processing an entire set of simulation paths together and renumbering the paths after each time step, correlation problems can be alleviated, allowing lower dimensional quasirandom points to be used. This greatly extends the range of simulations for which quasi-Monte Carlo can be used effectively, and, in the examples studied, has led to results far superior to either standard Monte Carlo or high dimensional quasi-Monte Carlo. Such techniques ought to be beneficial in such areas of stochastic simulation and Monte Carlo estimation as computational physics, finance, and neutron transport.

1.1 Stochastic Simulation

Stochastic simulation refers to the simulation of a stochastic process, in which the underlying variables are not simply deterministic, but are instead (discrete or continuous) random variables. Although the process is random, the result to be estimated is often a simple deterministic quantity, such as ground-state energy, expected profit, or neutron flux, for example.

A stochastic process whose future depends only on its current state, independent of past states, is said to be a Markov process, and it is with such processes which we will be concerned. In general terms, we shall deal with the application of quasi-Monte Carlo to Markov process, X_t, in s dimensions.

In order to simulate the process, X_t, over time, the time is discretized, so that we have $Y_j \approx X_{t_j}$ where $t_j = j\Delta t$ for $j = 0, \ldots, m$, $\Delta t = \frac{T}{m}$, and T is some maximum time of interest while m is the number of (equally spaced) time steps. (Note: It is easy to generalize to unequal time steps, but for simplicity equal time steps will be assumed throughout.)

We can symbolically represent the progress of the time-discrete Markov process, Y_j, in time using the following iterative rule:

$$Y_{j+1} = Y_j + g(Y_j, u_1, \ldots, u_d, t_j) \ , \ j = 0, \ldots, m-1 \tag{1}$$

This representation indicates that each successive position, Y_{j+1}, is obtained by moving from the preceding position, Y_j, by an amount which depends on Y_j, t_j, and d random input factors (without these factors the simulation would be deterministic). The details of the process are contained in the definition of g which depends on the particular problem to be solved. Typically the number of random inputs for each time step, d, equals the dimension of the process, s, but this is not required in general.

In the setting described above, given a function, f, the expected value of $f(X_T)$ can be estimated using the Monte Carlo method by simulating Y_j, the discretized process, N times (each simulation is called a 'path,' 'realization,' 'history,' or 'run') and then computing the estimate:

$$\hat{I}_N = \frac{1}{N} \sum_{i=1}^{N} f(Y_{i,m}) \tag{2}$$

where $Y_{i,j}$ represents the position of the i'th path after j time steps. This is a simple or 'crude' Monte Carlo estimate. As $N \to \infty$, $\hat{I}_N$ will approach $Ef(Y_m)$ at a rate proportional to $1/\sqrt{N}$ when random or pseudorandom points are used. Quasi-Monte Carlo aims to improve this rate and thereby reduce errors, as we shall see. (Note: The difference between $Ef(Y_m)$ and $Ef(X_T)$ is called discretization error. It is important in general, but not a subject of the present paper. In all the examples here, time steps are taken to be sufficiently small so that discretization errors are insignificant relative to Monte Carlo errors.)

2 High Dimensional Integral Method

The most direct way to apply quasi-Monte Carlo to a stochastic simulation is to interpret each single simulation path as a single evaluation of an implicitly defined, high dimensional, vector-valued function, G, the value of which is the endpoint of the path, Y_m, using the notation of the preceding section. (Note: whenever Y appears with just one subscript, as here, we are dealing with a single path and therefore the i subscript identifying the path is dropped.)

From Equation 1, we see that Y_m is a function of all of the random inputs, $u_1, \ldots, u_d$, at each time step along the path from Y_0 to Y_m. Therefore, we can symbolically write

$$Y_m = G(u_1, \ldots, u_d, u_{d+1}, \ldots, u_{2d}, \ldots, u_{md})$$

where the u_j's are numbered sequentially over the time steps, so for example the inputs for the second step are $u_{d+1}, \ldots, u_{2d}$, and G is 'evaluated' by generating a simulation path with the particular u_j's indicated.

Using the implicitly defined function G, we can then express the quantity estimated in Equation 2 as follows:

$$I = Ef(Y_m) = \int f(G(u_1, \ldots, u_{md}))\, du_1 \ldots du_{md}$$

where d is the number of random inputs required per time step and m is the number of time steps in each simulation path.

In the above, the u_j's are uniformly distributed. However, in practice they are usually transformed to have another probability distribution (e.g. Gaussian) in order to correctly simulate the stochastic process. Such transformations are included in the implicit definition of G since they are involved in the computation of the stochastic simulation path.

Quasi-Monte Carlo can be applied directly to the high dimensional integral above as follows:

$$\tilde{I}_N = \frac{1}{N} \sum_{i=1}^{N} f(G(q_i^{(1)}, \ldots, q_i^{(md)}))$$

where $q_i^{(k)}$ represents the k'th component of the i'th point from an md-dim quasirandom sequence (QRS). Therefore each individual simulation path involves the components of a single high dimensional quasirandom point. The estimate above is a simple integration estimate, and as such there is no problem with correlation errors from the quasirandom points.

This technique, called the high dimensional integral method, can be very effective, and has been successfully implemented for several applications including computer graphics (Heinrich and Keller [8]), geophysics (Shuhman [9]), and radiation transport (O'Brien [10], Sarkar and Prasad [11]).

Despite the successes listed above, the applications for which the high dimensional approach is effective are limited in practice because quasi-Monte Carlo loses its effectiveness as the number of dimensions rises. In very rough terms, the dimensions (md) should generally be less than about 30 in order for quasi-Monte Carlo to be significantly better than standard Monte Carlo. This is a generalization for a typical process with up to approximately a million paths. The actual effectiveness depends on the size of N as well as the particulars of the process being simulated. (See [1, 4, 5, 6, 7, 12] for further discussion.)

In certain situations, just a few time steps are required, and the high dimensional integral method works well, or else the importance of the higher dimensions is significantly less than that of the lower dimensions (for example, a financial simulation in which long term future changes in value have a relatively small impact on present values) and the high dimensional integral method may also work well. However, in a great many cases, one cannot count on having a

small number of time steps or having relatively unimportant later time steps. In such cases the limit of 30 dimensions is very quickly exceeded, for example a 2D process with 100 time steps per simulation requires 200 dimensions for the high dimensional integral method. This difficulty renders the high dimensional integral method ineffective for a large number of stochastic simulations.

3 Low Dimensional Methods

In seeking to find low dimensional methods for applying quasi-Monte Carlo to stochastic simulations, we hope to overcome the fundamental limitation of the high dimensional integral method discussed in the preceding section, which is essentially that it tends to run out of dimensions.

First, we examine the 'naive' approach. It is erroneous and should be avoided, as will be shown. We will then introduce the idea of renumbering which overcomes the problems of the naive approach and has shown itself to be highly effective in the examples considered here.

3.1 Naive Low Dimensional Method

A tempting, but wrong, approach to applying low dimensional quasi-Monte Carlo to stochastic simulation is to simply use m successive points from a d-dimensional quasirandom sequence (QRS) to generate each simulation path, using a total of mN quasirandom points for the entire collection of paths. Recall that d is the number of random inputs per time step, m is the number of time steps for each path, and N is the total number of paths simulated. This is the most likely method one would produce by simply replacing pseudorandom points with quasirandom points directly.

In effect, we would have the following:

$$\begin{aligned}\tilde{I}_N \;=\; & \frac{1}{N}\sum_{i=1}^{N} f\Big(G(q^{(1)}_{(i-1)m+1},\ldots,q^{(d)}_{(i-1)m+1},\\ & q^{(1)}_{(i-1)m+2},\ldots,q^{(d)}_{(i-1)m+2},\\ & q^{(1)}_{(i-1)m+3},\ldots,q^{(d)}_{(i-1)m+3},\\ & \ldots,q^{(1)}_{im},\ldots,q^{(d)}_{im})\Big)\end{aligned} \tag{3}$$

where $q_i^{(k)}$ represents the k'th component of the i'th quasirandom point in d-dimensions.

In Equation 3, the quasirandom points are used differently from the high dimensional approach in Section 2. Instead of a single md-dimensional quasirandom point for each simulation path, a set of m consecutive d-dimensional quasirandom points is used. This is a misuse of quasirandom points. It violates the rule, "one quasirandom point per integrand evaluation," which is implicit in the error estimates associated with quasirandom points based on discrepancy and the Koksma-Hlawka Inequality [3, 13]. As a result, correlations between successive quasirandom points cause large errors which do NOT disappear as N, the number of paths or histories, increases.

The problem comes from incorrectly treating quasirandom points as if they were simply 'better' pseudorandom points. They are highly correlated, and unless care is taken to avoid errors produced by these correlations, incorrect results will be the result. In the high dimensional integral method, the simulation is converted to an integral, for which quasirandom points can be applied directly, but in the present case the correlations are harmful.

Computational results in Example 1 will clearly show the poor results which can be expected using this erroneous approach.

3.2 Renumbering

We have found that by generating an entire set of simulation paths together and renumbering the paths after each time step, the correlation problems of the preceding section can be eliminated. This technique has been introduced previously by Lécot [14] and subsequently by Morokoff and Caflisch [15], and has been referred to variously as renumbering, reordering, or scrambling. (Note: This scrambling is different from for example the 'scrambled Halton sequence' since here it is the numbering of the paths which is scrambled.)

In the course of generating simulations, usually we would have

$$Y_{i,j+1} = Y_{i,j} + g(Y_{i,j}, u_1, \ldots, u_d, t_j)$$

but, we could just as well have

$$Y_{i,j+1} = Y_{k_i,j} + g(Y_{k_i,j}, u_1, \ldots, u_d, t_j)$$

where $\{k_i\}_{i=1}^N$ is a permutation of $\{1, \ldots, N\}$. In other words, after each time step the numbering of the N simulation paths is permuted.

This requires generating all N paths together, which often means that large amounts of memory are needed, a potentially serious drawback (see the note below).

On the other hand, computational results indicate that renumbering after each time step eliminates the correlation problems of the 'naive' method, leading

to faster convergence than either standard Monte Carlo or high dimensional quasi-Monte Carlo.

(Note: The large memory requirement can be a serious drawback in practice when, for example, millions of paths are generated. However, a potential solution to this problem is to generate the paths in smaller blocks of, say, a thousand, and then perform the renumbering within each block. Preliminary tests have been encouraging (Spanier [16]), but further study is indicated.)

Renumbering has been based on two different approaches which are described next.

3.2.1 Random Renumbering

For random renumbering, a random permutation of $\{1, \ldots, N\}$ is applied to the numbering of the simulation paths after each time step. This permutation changes, randomly, from time step to time step.

Without any renumbering, generating the entire set of paths together, we would have the following, using the notation of Section 2:

$$\begin{aligned} \tilde{I}_N &= \frac{1}{N}\sum_{i=1}^{N} f(G(q_i^{(1)}, \ldots, q_i^{(d)}, q_{i+N}^{(1)}, \ldots, q_{i+N}^{(d)}, \\ &\qquad q_{i+2N}^{(1)}, \ldots, q_{i+2N}^{(d)}, \ldots, q_{i+(m-1)N}^{(1)}, \ldots, q_{i+(m-1)N}^{(d)})) \end{aligned} \tag{4}$$

where the first N quasirandom points are used for the first time step of each path, the second N points for the second step, and so forth. As in Equation 3, each simulation path involves m quasirandom points in d dimensions. In Equation 3 the points used for a single path are consecutive while in Equation 4 they are separated by fixed increment N. However, in either of these cases these points are highly correlated when they are quasirandom, and as a result large errors are created.

With random renumbering we have instead the following:

$$\begin{aligned} \tilde{I}_N &= \frac{1}{N}\sum_{i=1}^{N} f(G(q_i^{(1)}, \ldots, q_i^{(d)}, q_{k_{i,1}+N}^{(1)}, \ldots, q_{k_{i,1}+N}^{(d)}, \\ &\qquad q_{k_{i,2}+2N}^{(1)}, \ldots, q_{k_{i,2}+2N}^{(d)}, \ldots, q_{k_{i,m-1}+(m-1)N}^{(1)}, \ldots, q_{k_{i,m-1}+(m-1)N}^{(d)})) \end{aligned} \tag{5}$$

where $\{k_{i,j}\}_{i=1}^{N}$ is the random permutation of $\{1, \ldots, N\}$ for the j'th time step.

In Equation 5, each simulation path again involves m quasirandom points in d dimensions. However, unlike in Equations 3 and 4, the random renumbering prevents any simple relationships between the points used for any single path from developing. This has the effect of eliminating the correlations between these points and therefore eliminating the errors caused by such correlations.

Essentially, by randomly renumbering the paths, the quasirandom points $q_i, q_{i+N}, q_{i+2N}, \cdots, q_{i+(m-1)N}$ which are correlated and therefore produce large errors, are replaced by points $q_i, q_{k_{i,1}+N}, q_{k_{i,2}+2N}, \cdots, q_{k_{i,m-1}+(m-1)N}$ which are uncorrelated when N is large and each $k_{i,j}$ is random.

3.2.2 Binary Renumbering

For binary renumbering, the region in which simulation paths travel is split into L levels in each coordinate direction, where L is a power of two. Then a fixed numbering of the L^s resultant subregions, in s-dimensions, is produced which has the property that all of the subregions contained within a cruder binary splitting of the region are numbered in contiguous blocks. (For example, in 2D all of the subregion in the first quadrant would precede all the subregions in the second quadrant.) With $L = 8$ in 2D we might have the following numbering:

1	2	5	6	17	18	21	22
3	4	7	8	19	20	23	24
9	10	13	14	25	26	29	30
11	12	15	16	27	28	31	32
33	34	37	38	49	50	53	54
35	36	39	40	51	52	55	56
41	42	45	46	57	58	61	62
43	44	47	48	59	60	63	64

This needs to be done just once. Then after each time step, paths are renumbered according to which subregion they fall within (randomizing within subregions). This is done so that paths which are close together in space will likely be numbered close together when they are renumbered. In Section 5.2 we will examine why this is a desirable property, in terms of variation.

Computational results indicate that this renumbering is often preferable to random renumbering when it is practical, as discussed further in Section 5.2.

4 Application: Randomized 2D Lotka-Volterra

As an example of the application of quasi-Monte Carlo to a stochastic simulation, we consider in this section a common population model of two interacting species (e.g. predator/prey) with an extra random term associated with random factors (e.g. weather conditions, randomness in births and deaths), as described

by Kloeden and Platen [17]. The randomness is included in the form of multiplicative Gaussian noise which has the desirable property of scaling random effects to current population levels.

The system is governed by the following stochastic differential equations:

$$dX_t^i = X_t^i \left(a^i + \sum_{j=1}^{2} b^{i,j} X_t^j \right) dt + \sigma^i X_t^i dW_t^i \ , \quad i = 1, 2$$

with X_0^1, X_0^2 = exact init populations, a^i = birth/death rates, $b^{i,j}$ = interaction coefficients, σ^i = sensitivities to random factors.

Example 1 (Lotka-Volterra) *We examine numerical results for the randomized 2D Lotka-Volterra system described above with the following coefficients:*

$$a = \begin{pmatrix} -.3 \\ -.3 \end{pmatrix} \quad b = \begin{pmatrix} .01 & .03 \\ -.04 & .02 \end{pmatrix}$$
$$\sigma = \begin{pmatrix} .08 \\ .14 \end{pmatrix}$$

and with initial conditions:

$$X_0^1 = 3.0 \ , \quad X_0^2 = 5.0$$

We wish to estimate $Ef(X_T)$ for the following test function:

$$f(X_T) = (X_T^1)^2 + (X_T^2)^2 \ \ at \ T = 5$$

Five different methods for estimating $Ef(X_T)$ are compared. The particular quasirandom sequence used is the Niederreiter sequence, named after its developer [5]. Methods compared:

1. *Naive low dimensional quasirandom*

   2D Niederreiter sequence, $m = 80$ time steps, 80 points per path, no renumbering (NOT expected to converge.)

2. *High dimensional integral method (quasirandom)*

   80D Niederreiter sequence, $m = 40$ time steps, one point per path.

3. *Standard Monte Carlo*

   Pseudorandom sequence, $m = 80$ time steps.

4. *Random Renumbering (quasirandom)*

2D Niederreiter sequence, $m = 80$ time steps, 80 points per path, random renumbering after each time step.

5. *Binary Renumbering (quasirandom)*

   2D Niederreiter sequence, $m = 80$ time steps, 80 points per path, binary renumbering ($L = 256$) after each time step.

In Figure 1, the results for each of these methods are plotted. Each method is computed for 50 repetitions at each of several different values of N, and then the root-mean-square error (e) over these repetitions is plotted against the number of paths (N). A log-log scale is used for the axes so that when the error is of the form $e = bN^a$, the plotted line will appear as a straight line with slope a, the convergence rate. For standard Monte Carlo we expect a to be approximately equal to -0.5, while for quasi-Monte Carlo we expect it to be lower.

In addition, for a more realistic comparison of the methods, in Figure 2 the errors are plotted against the average amount of CPU time (t) used in computing each estimate. A log-log scale again is used for easy comparison with the results in Figure 1.

Note: Since an exact solution for this problem was not available. An estimated solution using standard Monte Carlo with $N = 5,000,000$ was used in computing root-mean-square errors. See [4, 12] for a further discussion of error measurement techniques for quasi-Monte Carlo.

We see from the plotted results that standard Monte Carlo converges at its expected rate ($a \approx -0.5$).

The high dimensional integral method (quasirandom) converges at a slightly faster rate but with larger errors than standard Monte Carlo. It is not surprising that for 80 dimensions quasi-Monte Carlo is worse than standard Monte Carlo. In fact it is somewhat surprising that the results are as good as they are in this case.

The low dimensional quasi-Monte Carlo method without renumbering, as expected, clearly does not converge to the correct solution. This is clear from the horizontal line in Figure 1, which indicates that the error is not decreasing as N increases.

On the other hand, with renumbering, the low dimensional quasi-Monte Carlo methods do appear to converge. With random renumbering, the convergence rate is only slightly improved but the errors are reduced (through the constant factor, denoted b above). With binary renumbering, we see the greatest improvement, with a significant reduction in the convergence rate to $a = -0.89$ and correspondingly low levels of error.

The plots against CPU time show that while binary renumbering involves an additional amount of start-up time compared with the other methods, this extra time becomes relatively insignificant once N reaches moderately high values.

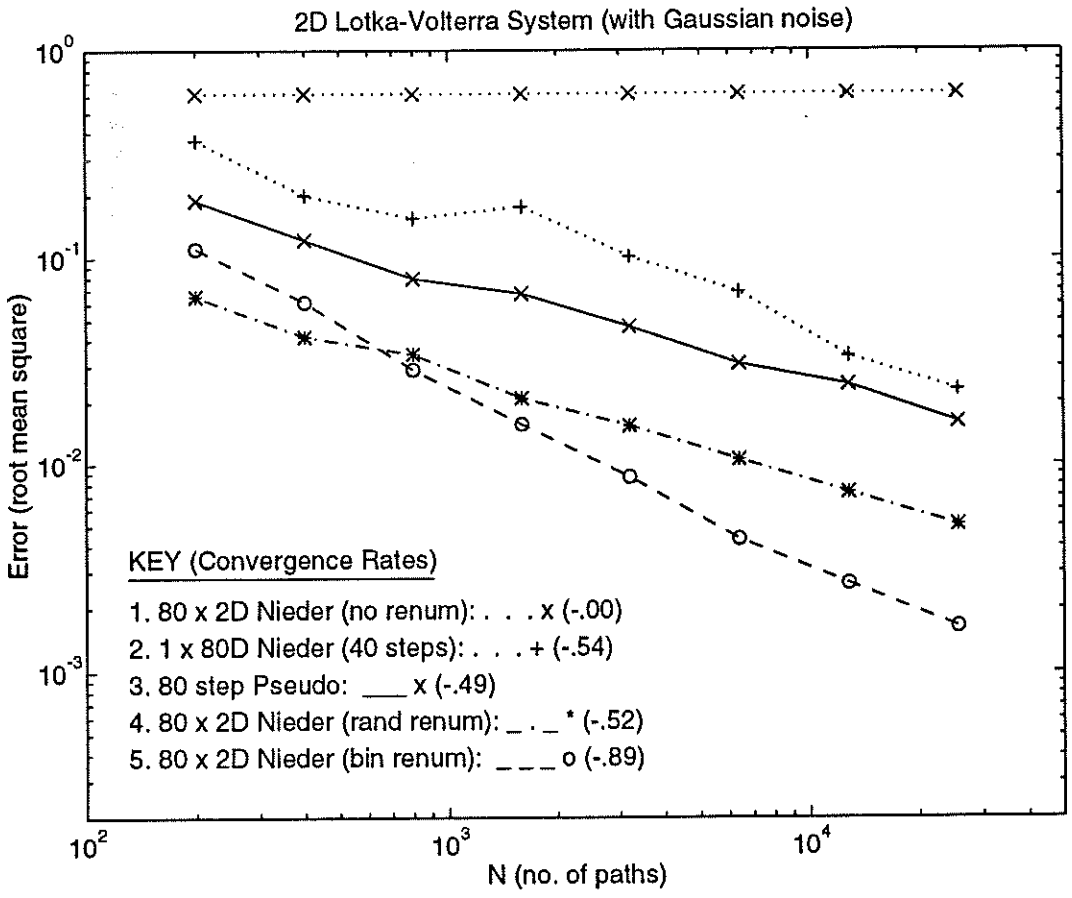

Figure 1: Results, 50 reps, Example 1.

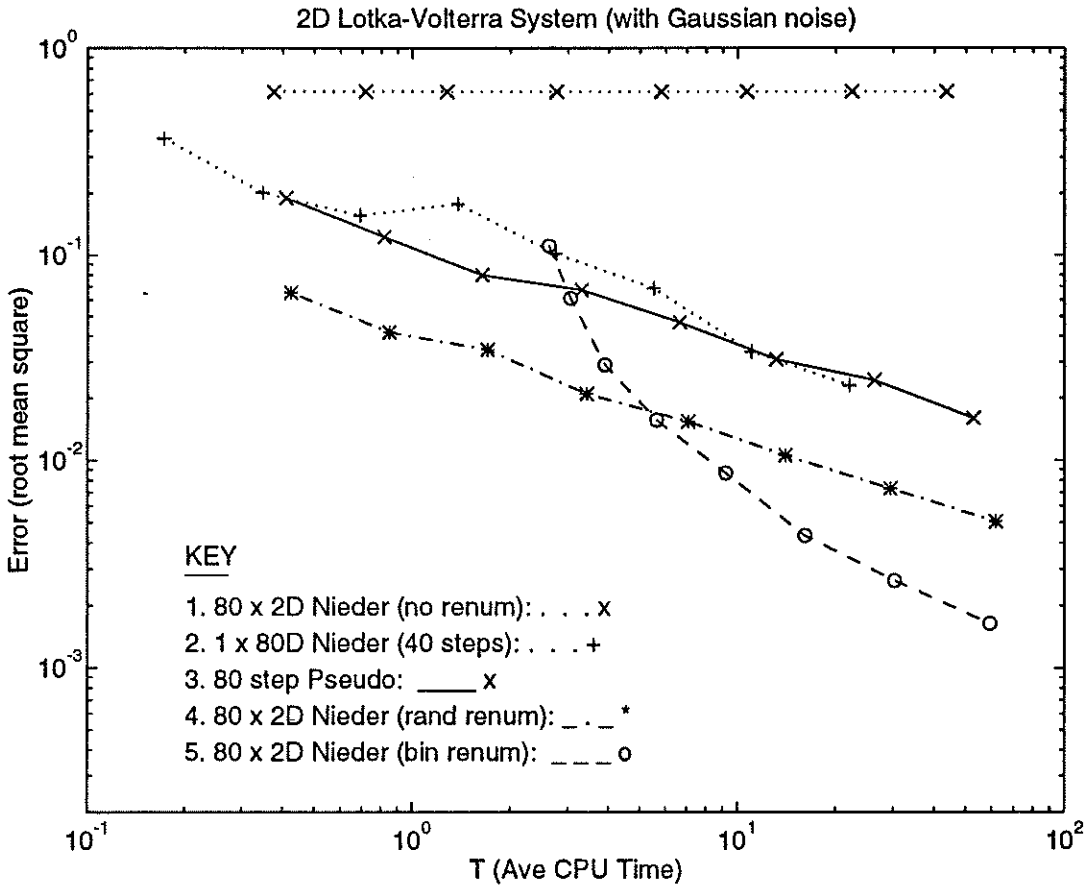

Figure 2: Results, 50 reps, Example 1.

5 Further Analysis of Renumbering

In this section, the motivation for renumbering will be discussed in some more detail. While we have not yet succeeded in proving that low dimensional quasi-Monte Carlo with renumbering converges to the exact solution in all cases of stochastic simulation, the reasons for renumbering's effectiveness are clear. The key is the ability to eliminate harmful correlations, as discussed in Section 3.2, by renumbering, either randomly or in a more orderly fashion as in the case of binary renumbering.

The arguments presented here attempt to provide a more complete examination of renumbering as well as a framework within which future proofs may be forthcoming.

For simplicity, consider the next to last step of a set of N simulation paths with renumbering. Let $p_i = Y_{i,m-1}$, using the notation introduced in Section 1.1. Then, one could theoretically estimate $I = Ef(Y_m)$ as follows:

$$I^* = \frac{1}{N}\sum_{i=1}^{N}\int f(p_i + g(p_i, u_1, \ldots, u_d, t_{m-1}))\, du_1 \ldots du_d$$

using Equation 1 with $j = m-1$. This estimate is not in an easily computable form, but it will serve as our starting point.

Next, define "selection" function

$$s(u) = \sum_{i=1}^{N} p_i \, \mathcal{X}\left\{\frac{i-1}{N} \le u < \frac{i}{N}\right\} \quad , \quad 0 \le u < 1$$

where $\mathcal{X}$ is a characteristic, or indicator, function which equals unity within the indicated set and zero in its complement. For u uniformly distributed between zero and one, this function has an equal probability of being assigned any particular p_i as its (vector) value.

Then, we have

$$I^* = \int f\left(s(u_1) + g(s(u_1), u_2, \ldots, u_{d+1}, t_{m-1})\right)\, du_1 \ldots du_{d+1}$$

This is a simple integral which can be estimated using quasi-Monte Carlo directly as follows:

$$\hat{I}_N = \frac{1}{N}\sum_{i=1}^{N} f\left(s(q_i^{(1)}) + g(s(q_i^{(1)}), q_i^{(2)}, \ldots, q_i^{(d+1)}, t_{m-1})\right) \tag{6}$$

In effect this equation says that we should use the first component of each quasirandom point, $q_i^{(1)}$, to select one of the p_i's and then proceed one time step

from there using the remaining components $q_i^{(2)}, \ldots, q_i^{(d+1)}$. This is therefore a $(d+1)$-dimensional integral. One obvious drawback of the above estimate is that it is likely to include repetitions of some of the p_i's, which could be randomly selected more than once, and exclusion of others, which has the unpleasant effect of increasing errors.

One simple way to avoid repetitions is to simply not allow points to be selected more than once, by either adjusting $s(u)$ after each selection, or rejecting all repeat selections. This can be effective and leads to a type of method which has been dubbed 'correlated continuation' [12] by the present author.

However, there is another way to handle this situation, based on the Hammersley sequence, which justifies taking the paths in order and simplifies both the analysis and implementation. This is discussed next.

5.1 Use of the Hammersley Sequence

Given any d-dimensional QRS $(q_i^{(1)}, \ldots, q_i^{(d)})$, $i = 1, 2, \ldots$, we define the corresponding $d+1$-dimensional Hammersley sequence for fixed N as follows:

$$h_i = \left(\frac{i-1/2}{N}, q_i^{(1)}, \ldots, q_i^{(d)} \right) \quad , \quad i = 1, \ldots, N$$

(Note: This is sometimes termed a 'point set' to emphasize that the values of the points depend on N.) This sequence has the same discrepancy as the original QRS, while increasing the dimensions by one [3, 18].

Now, applying this sequence in Equation 6 has the very useful consequence that each p_i is selected exactly once in order. This eliminates any possible repetitions and exclusions, and it produces the following simple estimate:

$$\hat{I}_N = \frac{1}{N} \sum_{i=1}^{N} f\left(p_i + g(p_i, q_i^{(1)}, \ldots, q_i^{(d)}) \right) \tag{7}$$

The procedure described above can be iterated over every time step, producing the following estimate, which is the same as Equation 4:

$$\begin{aligned} \tilde{I}_N \quad = \quad & \frac{1}{N} \sum_{i=1}^{N} f(G(q_i^{(1)}, \ldots, q_i^{(d)}, q_{i+N}^{(1)}, \ldots, q_{i+N}^{(d)}, \\ & q_{i+2N}^{(1)}, \ldots, q_{i+2N}^{(d)}, \ldots, q_{i+(m-1)N}^{(1)}, \ldots, q_{i+(m-1)N}^{(d)})) \end{aligned}$$

As discussed earlier, however, this estimate is problematic because of the correlations between $q_i, q_{i+N}, q_{i+2N}, \ldots, q_{i+(m-1)N}$ without renumbering. Referring back to Equation 7, the problem stems from p_i being correlated with q_i, which is a natural consequence of the orderly way in which paths are normally generated.

Renumbering eliminates the correlations between p_i and q_i. It can be inserted into the analysis as follows:

Let the selection function be permuted as follows:

$$\tilde{s}(u) = \sum_{i=1}^{N} p_{k_i} \mathcal{X}\left\{\frac{i-1}{N} \leq u < \frac{i}{N}\right\} \quad , \quad 0 \leq u < 1 \tag{8}$$

where $\{k_i\}_{i=1}^{N}$ is a permutation of $\{1, \ldots, N\}$. Then as before we would have

$$I^* = \int f\left(\tilde{s}(u_1) + g(\tilde{s}(u_1), u_2, \ldots, u_{d+1}, t_{m-1})\right) du_1 \ldots du_{d+1} \tag{9}$$

Applying the Hammersley sequence to this gives us

$$\hat{I}_N = \frac{1}{N} \sum_{i=1}^{N} f\left(p_{k_i} + g(p_{k_i}, q_i^{(1)}, \ldots, q_i^{(d)}, t_{m-1})\right) \tag{10}$$

and then iterating over the time steps, using a new permutation for each time step, gives us

$$\begin{aligned} \tilde{I}_N \quad = \quad & \frac{1}{N} \sum_{i=1}^{N} f(G(q_i^{(1)}, \ldots, q_i^{(d)}, q_{k_{i,1}+N}^{(1)}, \ldots, q_{k_{i,1}+N}^{(d)}, \\ & q_{k_{i,2}+2N}^{(1)}, \ldots, q_{k_{i,2}+2N}^{(d)}, \ldots, q_{k_{i,m-1}+(m-1)N}^{(1)}, \ldots, q_{k_{i,m-1}+(m-1)N}^{(d)})) \end{aligned}$$

which is the same as Equation 5.

The key difference, as discussed earlier, between this and the previous estimate is that instead of using the correlated points $q_i, q_{i+N}, \ldots$ in generating each single path, the points used for each path are instead $q_i, q_{k_{i,1}+N}, q_{k_{i,2}+2N}, \ldots$, which should be uncorrelated, for N sufficiently large, due to the renumbering.

A complete proof of convergence would require proving conclusively that renumbering does indeed eliminate any correlations of the type that will cause errors. While it seems intuitively true, and the computational results presented here support such a claim, a rigorous proof remains to be completed, hopefully in the not too distant future.

In the case of random renumbering such a proof would likely rely on the independence of the random permutations with respect to the quasirandom sequence. For binary renumbering, an argument based entirely on discrepancy is probably necessary. In fact, Morokoff and Caflisch [6, 15] have proven that estimates for simulations of the 1D Heat Equation using a 1D QRS and left to right renumbering, as described in Section 5.2, converge exactly to the correct solutions.

In effect with renumbering, we have broken up an md-dimensional integral into a series of lower dimensional integrals stacked together. The improvement over standard Monte Carlo for these estimates comes from each d-dim time step being integrated more accurately,

5.2 Binary Renumbering as a Variation Reduction Technique

We now discuss the argument for binary renumbering and show how its improved effectiveness when compared with random renumbering can be explained by treating it as a variation reduction technique.

Referring to the preceding section, the accuracy of $\hat{I}_N$ from Equation 10 in estimating

$$I^* = \int f\left(\tilde{s}(u_1) + g(\tilde{s}(u_1), u_2, \ldots, u_{d+1}, t_{m-1})\right) du_1 \ldots du_{d+1}$$

from Equation 9 using quasi-Monte Carlo depends, among other things, on the variation of the integrand.

A large part of the variation of the integrand above can generally be expected to come from $\tilde{s}(u)$, which is defined in Equation 8, since this function is in effect a vector-valued step function which jumps between values corresponding to random path locations which can be widely spread out over the range of possible paths for the underlying stochastic process.

Binary renumbering is nothing more than an attempt to reduce this large source of variation by using a renumbering scheme in which successively numbered paths will generally be close together. In terms of $\tilde{s}$, this means that successive values of $\tilde{s}$, as a function of u, will be close together. This has the effect of reducing the size of the jumps taken by $\tilde{s}(u)$ and therefore reducing the variation of the integrand as a whole.

For one dimension, this is accomplished simply by renumbering the paths from left to right after each time step. (As mentioned above, Morokoff and Caflisch have proven that for simulations of the 1D Heat Equation using a 1D QRS with left to right renumbering, estimates will converge to exactly the correct solutions.)

For more than one dimension, there is no longer an obvious best way to proceed. However, the method introduced in Section 3.2.2 as binary renumbering appears to be a practical way to accomplish this without incurring large computational costs. As discussed in Section 3.2.2, for binary renumbering the range of possible simulation paths is divided along each coordinate direction into L levels where L is a power of two. In s dimensions, this breaks up the region into a total of L^s subregions.

As the simulation proceeds, paths are stored according to which subregion they fall within when they are computed for each time step. Then, following each time step, the paths are renumbered by cycling though all of the subregions in a fixed order, as described in Section 3.2.2. Paths within the same subregion are randomly ordered. This can all be done in nearly order N time. (It is asymptotically order $N \log N$ due to the randomizing which is done within subregions.)

In the computational examples, we see that binary renumbering clearly outperforms random renumbering both in terms of convergence rate and in terms of actual error levels on a consistent basis. Therefore, it appears that the variation reduction is effective.

One potential disadvantage of binary renumbering is that it is dependent on the range of the particular process to be simulated, while random renumbering is more problem independent. For problems with complicated ranges, an adaptive method based on the binary renumbering strategy described here may be worthwhile.

6 Application: 3D Ornstein-Uhlenbeck Process

We now consider the three dimensional Ornstein-Uhlenbeck process with reflecting boundaries. This is a very common stochastic diffusion process in which the drift rate is a linear function of position, and the diffusion rate is uniform in all directions. It is discretized for this example using a simple Euler scheme. The system is governed by the following vector stochastic differential equation:

$$dX_t = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} X_t \, dt + dW$$

with reflecting boundaries at ± 3 in each component.

Example 2 (Ornstein-Uhlenbeck) *We examine numerical results for the 3D Ornstein-Uhlenbeck process with the following coefficients:*

$$A = \begin{pmatrix} 0.3 & 0.2 & 0.1 \\ -0.1 & 0.3 & 0.2 \\ -0.1 & -0.2 & 0.3 \end{pmatrix}$$

The initial conditions are probabilistic as follows:

$$X_0 \sim \exp(-(x^2 + y^2 + z^2))$$

where x, y, z are the three components of X_t.

We estimate $Ef(X_T)$ where

$$f(X_T) = \sqrt{(x+3)(y+3)(z+3)}$$

at $T = 5$, using $\Delta t = 0.05$ ($m = 100$ time steps).

The following three methods are compared:

1. *Standard Monte Carlo*

   Pseudorandom sequence.

2. *Random Renumbering (quasirandom)*

   3D Niederreiter sequence, 100 time steps, 100 points per path, random renumbering after each time step.

3. *Binary Renumbering (quasirandom)*

   3D Niederreiter sequence, 100 time steps, 100 points per path, binary renumbering ($L = 16$) after each time step.

Figure 3 is a log-log plot of the root-mean-square error over 100 repetitions versus N (the number of simulation paths), and Figure 4 is a log-log plot of the error versus CPU time. See Example 1 for a further explanation of the graphs. As in Example 1, a standard Monte Carlo estimate with large N (in this case $N = 2,500,000$) is used to obtain an 'exact' solution with which the estimates are compared when computing root-mean-square errors.

The results confirm that renumbering is effective for this example. In the case of random renumbering, minimal gain in convergence rate is observed, but the errors are reduced in an absolute sense when compared with standard Monte Carlo.

For binary renumbering, as expected, superior results are observed. There is an improvement in both the convergence rate as well as the overall level of errors. In this example, a relatively small number of levels is used ($L = 16$) for the binary renumbering. Therefore, only a small start-up time is evident in the plot against CPU time for binary renumbering.

7 Application: Hydrogen Atom (Diffusion MC)

Diffusion Monte Carlo (DMC) is a method for estimating the energies of quantum mechanical systems by simulating their wavefunctions. It is part of a more general class of methods collectively called Green's Function Monte Carlo (GFMC) [19, 20, 21]. In DMC, the Schrödinger Equation is transformed into a diffusion equation in imaginary time. The ground state energy is equal to the average path integral of a growth factor over stochastic paths representing electron positions. Estimates are computed using a Feynman-Kac type formula.

Here results are presented for a simple example involving estimates of the ground state energy of Hydrogen. A more complete explanation of the application of quasi-Monte Carlo to Diffusion Monte Carlo and the meaning of 'continuation' in the example below may be found in [12]. The basic idea of

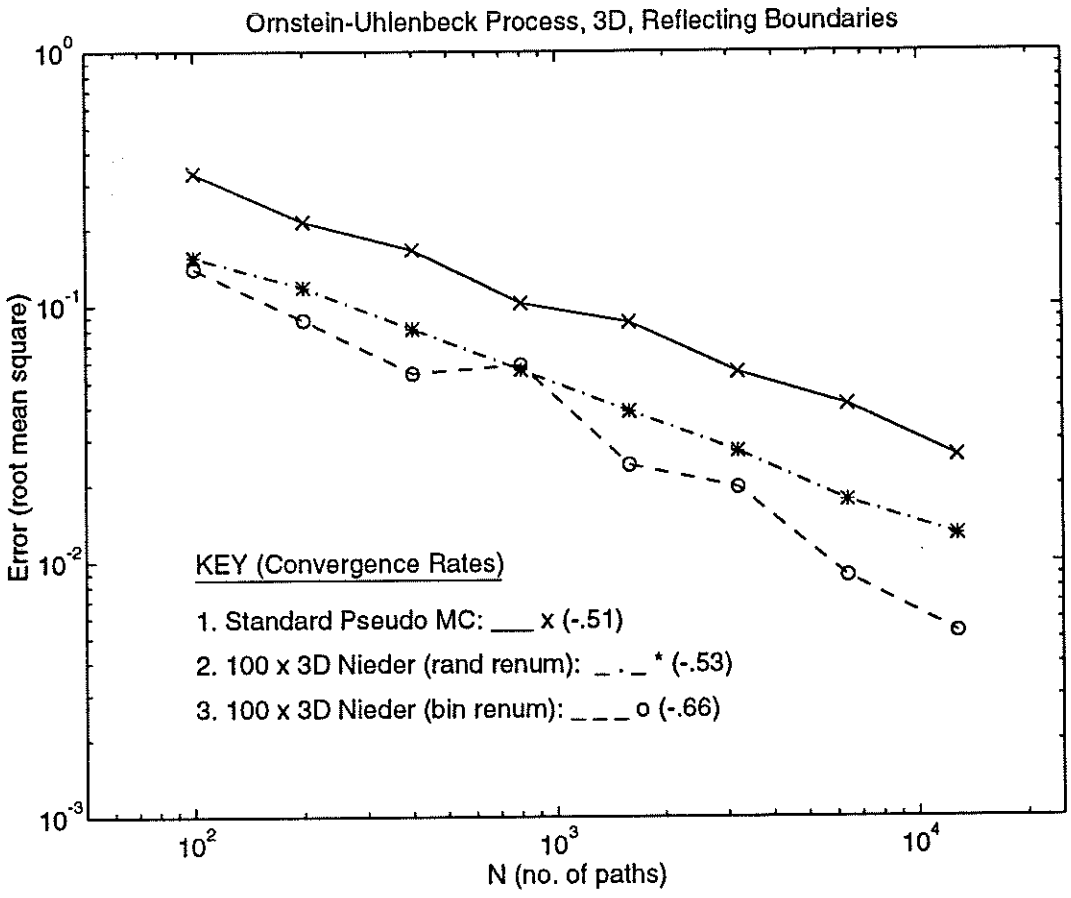

Figure 3: Results, 100 reps, Example 2.

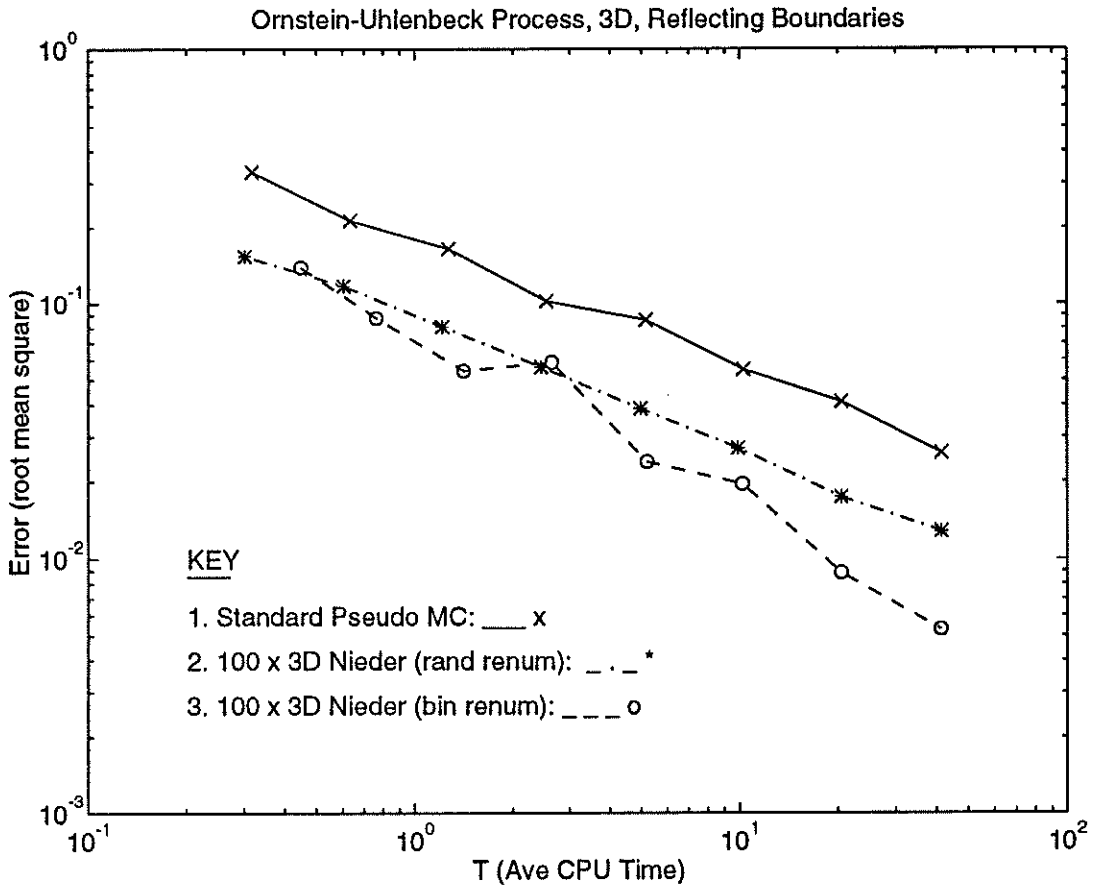

Figure 4: Results, 100 reps, Example 2.

continuation, as discussed in detail in [12], is to begin with standard pseudo-random Monte Carlo paths which are followed for a sufficiently long time using sufficiently small time steps so that time-discretization errors and ground-state convergence errors are insignificant relative to the statistical or 'Monte Carlo' uncertainty. Then quasi-Monte Carlo is used to reduce the 'Monte Carlo' error from levels of order $O(N^{-1/2})$ to levels between $O(N^{-1/2})$ and $O(N^{-1})$. As is clear in the following example, this can be a significant improvement.

Example 3 (Hydrogen Atom) *The ground state energy of Hydrogen, equal to -0.5 a.u., is estimated using Diffusion Monte Carlo. An importance (or trial) function of the form*

$$\phi_T(r) \sim e^{-r+\lambda r^2}$$

is used with $\lambda = 0.01$ *in order to smooth out the singularity at the nucleus while avoiding a trivial zero-variance condition.*

The simulations here use time steps of size $h = 0.2$ *with a third order accurate discretization (Helfand and Greenside [22]). This discretization allows larger time steps to be used but also requires two random variables per dimension per time step (i.e. 6 random variables per time step.)*

Results are compared for the following cases:

1. *Standard Monte Carlo*

   Using pseudorandom numbers, a set of 50 random walkers is followed, recording positions after every 20 time steps, until N positions have been accumulated. Then, the energy is estimated over this set of N positions.

2. *High dimensional continuation (pseudo/quasirandom)*

   Standard MC is used to accumulate N positions, as in case 1 above. Then, each position is advanced or 'continued' for an additional 4 time steps using a single point from a 24D Niederreiter sequence. The energy is estimated over this new set of N positions.

3. *Low dimensional continuation with renumbering (pseudo/quasirandom)*

   Standard MC is used to accumulate N positions, as above. Then, each position is advanced for an additional 20 time steps using 20 points from a 6D Niederreiter sequence, one for each time step. Binary renumbering ($L = 32$) *is performed following each of the 'continuation' time steps.*

4. *Low dimensional quasirandom with renumbering (quasirandom)*

   A set of N random walkers is followed for 70 time steps, without accumulating positions, but rather, using the final position of each walker. Each path requires 70 points from a 6D Niederreiter sequence. Binary renumbering ($L = 32$) *is performed following every time step.*

Results for this example are presented in the form of log-log plots, as in the first two examples. Figure 5 is a plot of the root-mean-square error versus N, and Figure 6 is a plot of the error versus CPU time. The results are based on 100 repetitions.

The results in Figures 5 and 6 indicate that all three of the methods which use quasirandom points produce better results than standard Monte Carlo for this example. The fourth method converges most rapidly with respect to N, which is not surprising since it is the only totally quasi-Monte Carlo method compared. However, the plots against CPU time show that the third method, low dimensional continuation, may be preferable in terms of CPU time. This can be explained by the higher efficiency of using just 50 random walkers and accumulating new positions every 20 time steps as they move. The second method, not surprisingly, is not as effective as the last two because the dimensions are higher, which causes quasi-Monte Carlo to lose some of its advantage, as discussed in Section 2.

The application of these methods to more challenging quantum mechanical systems deserves further study,

8 Conclusions

The application of quasi-Monte Carlo methods to stochastic simulation is a largely unexplored area with great future potential. The concept of renumbering overcomes the common problem of 'running out of dimensions,' which is a fundamental limitation of the high dimensional integral method, as discussed in Section 2, by allowing lower dimensions to be reused in successive time steps. Renumbering has the effect of eliminating harmful correlations which cause large errors when dimensions are reused without any renumbering. Although a rigorous proof that such correlations are completely eliminated has not yet been completed, the computational results presented here show clear evidence of the usefulness of renumbering in the examples studied.

Further, the arguments in Section 5.2, as well as the computational results in the examples, show that a more orderly form of renumbering, called binary renumbering, can be significantly more effective than simple random renumbering. This can be explained in terms of variation by treating binary renumbering as a form of variation reduction, which reduces the quasi-Monte Carlo errors.

There is a great deal more to be studied and proven about the application of quasi-Monte Carlo to stochastic simulation, but the results so far have been encouraging and it appears that such methods may ultimately prove to be extremely worthwhile.

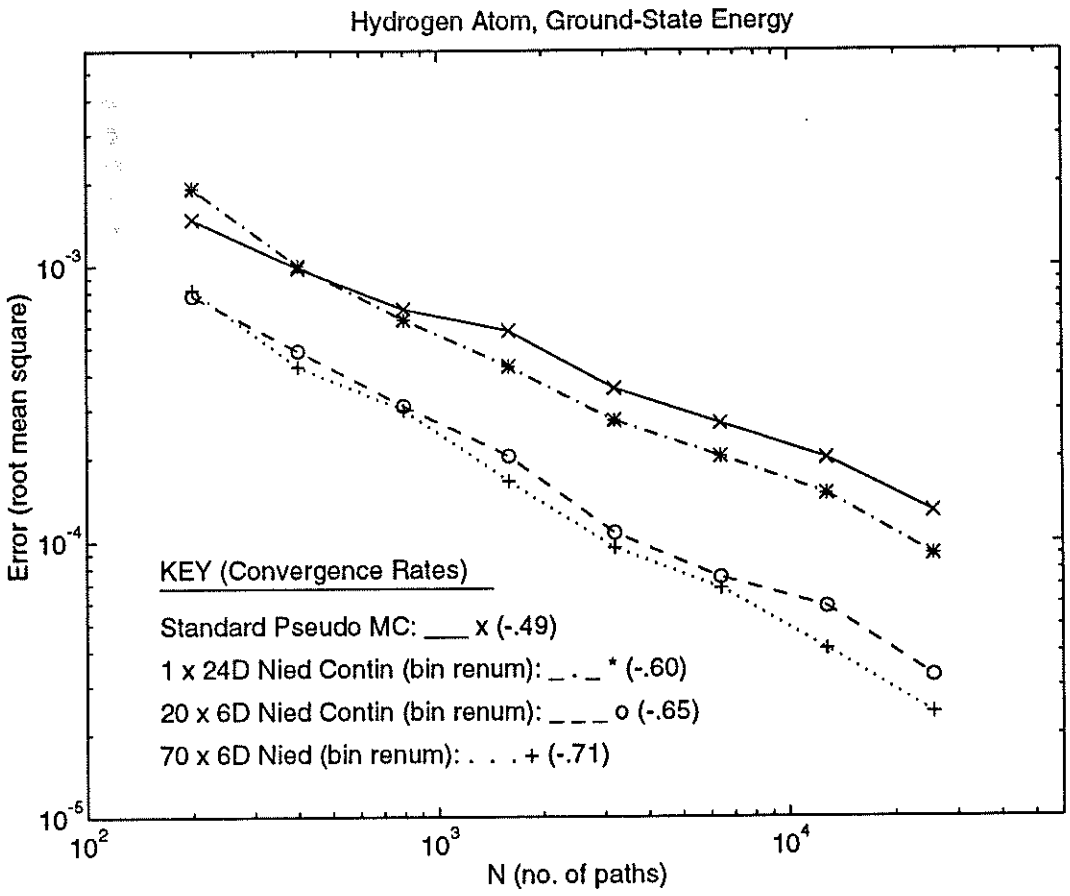

Figure 5: Results, 100 reps, Example 3.

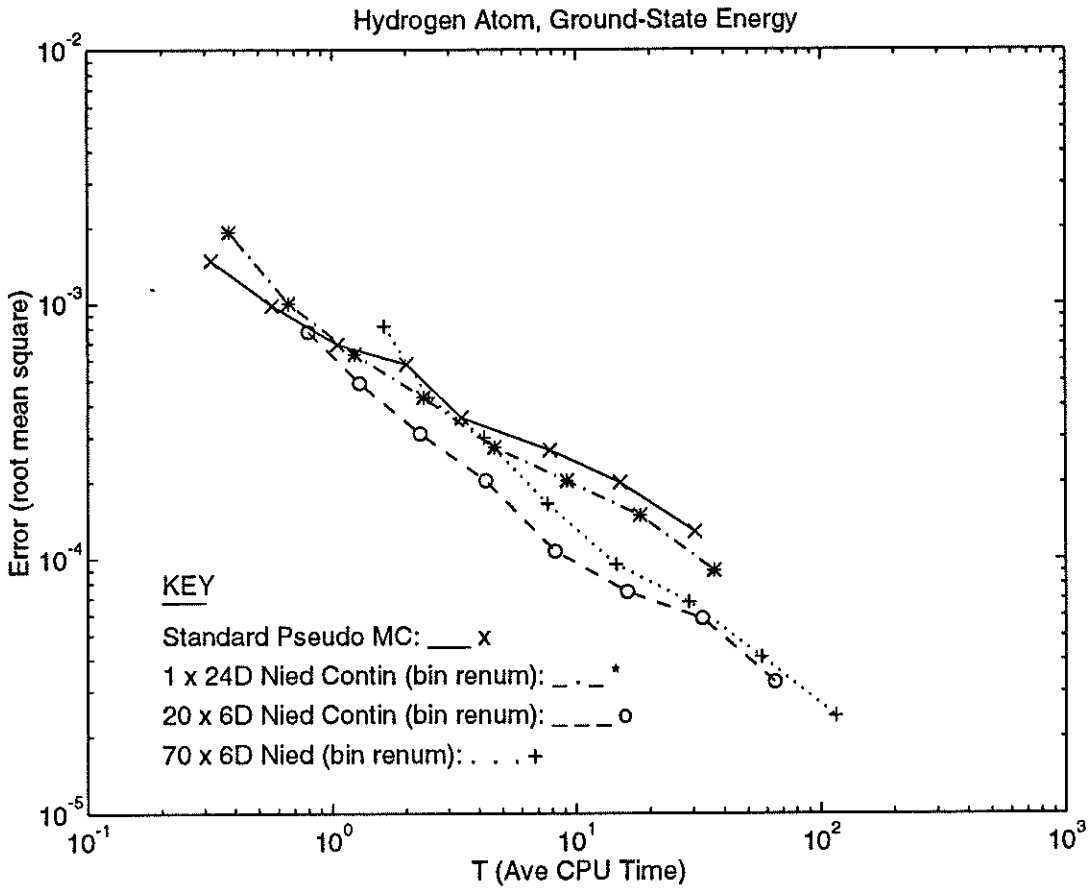

Figure 6: Results, 100 reps, Example 3.

References

[1] William H. Press and Saul A. Teukolsky. Quasi- (that is, sub-) random numbers. *Computers in Physics*, pages 76–79, 1989.

[2] M. Berblinger and C. Schlier. Monte Carlo integration with quasi-random numbers: Some experience. *Computer Physics Communications*, 66:157–166, 1991.

[3] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods.* SIAM, Philadelphia, 1992.

[4] Bradley Moskowitz and Russel E. Caflisch. Smoothness and dimension reduction in quasi-Monte Carlo methods. *Mathematical and Computer Modelling*, submitted, March 1994.

[5] Paul Bratley, Bennett L. Fox, and Harald Niederreiter. Implementation and tests of low-discrepancy sequences. *ACM Transactions on Modeling and Computer Simulation*, 2(3):195–213, July 1992.

[6] W.J. Morokoff. *Quasi-Monte Carlo Methods for Numerical Integration and Simulation.* PhD thesis, New York University, May 1990.

[7] J. Spanier and E.H. Maize. Quasi-random methods for estimating integrals using relatively small samples. *SIAM Review*, to appear, 1994.

[8] Stefan Heinrich and Alexander Keller. Quasi-monte carlo methods in computer graphics, part II: The radiance equation. Technical Report 243/94, Fachbereich Informatik, AG Numerische Algorithmen, Universität Kasierslautern, 1994.

[9] B.V. Shuhman. Application of quasirandom points for simulation of gamma radiation transfer. *Progress in Nuclear Energy*, 24:89–95, 1990.

[10] D.M. O'Brien. Accelerated quasi Monte Carlo integration of the radiative transfer equation. *J. Quant. Spectrosc. Radiat. Transfer*, 48(1):41–59, 1992.

[11] P.K. Sarkar and M.A. Prasad. A comparative study of pseudo and quasi random sequences for the solution of integral equations. *Journal of Computational Physics*, 68:66–88, 1987.

[12] B. Moskowitz. *Application of Quasi-Random Sequences to Monte Carlo Methods.* PhD thesis, University of California, Los Angeles, September 1993.

[13] L. Kuipers and H. Niederreiter. *Uniform Distribution of Sequences.* Wiley, New York, 1974.

[14] C. Lécot. A quasi-Monte Carlo method for the Boltzmann equation. *Mathematics of Computation*, 56(194):621–644, April 1991.

[15] W. Morokoff and R.E. Caflisch. A quasi-Monte Carlo approach to particle simulation of the heat equation. *SIAM Journal on Numerical Analysis*, to appear, 1993.

[16] J. Spanier, June 1994. presentation at Las Vegas Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing.

[17] P.E. Kloeden and E. Platen. *Numerical Solution of Stochastic Differential Equations*. Springer-Verlag, Berlin; New York, 1992.

[18] H. Niederreiter. Quasi-Monte Carlo methods and pseudo-random numbers. *Bulletin of the AMS*, 84(6), November 1978.

[19] M.H. Kalos. Monte Carlo calculations of the ground state of three- and four- body nuclei. *Physical Review*, 128(4):1791–1795, November 1962.

[20] P.J. Reynolds, J. Tobochnik, and H. Gould. Diffusion quantum Monte Carlo. *Computers in Physics*, pages 662–668, Nov/Dec 1990.

[21] C.A. Traynor, J.B. Anderson, and B.M. Boghosian. A quantum Monte Carlo calculation of the ground state energy of the hydrogen molecule. *Journal of Chemical Physics*, 94(5):3657–3664, March 1991.

[22] H.S. Greenside and E. Helfand. Numerical integration of stochastic differential equations - II. *The Bell System Technical Journal*, pages 1927–1941, October 1981.