UCLA
COMPUTATIONAL AND APPLIED MATHEMATICS

Quasirandom Diffusion Monte Carlo

Bradley Moskowitz

December 1994

CAM Report 94-39

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

Quasirandom Diffusion Monte Carlo

Bradley Moskowitz *
Mathematics Department, UCLA

October 20, 1994

Abstract

Diffusion Monte Carlo is a common method for estimating the properties of quantum mechanical systems by computing averages over sets of random walk simulations. We have found that by using quasirandom sequences of points in place of random or pseudorandom points in generating the simulation paths, we are able to obtain improved convergence rates and consequently reduced Monte Carlo errors for Diffusion Monte Carlo. Computational results are presented for a three dimensional harmonic oscillator and the Helium atom.

A key element in successfully applying the quasirandom sequence is a recently developed technique involving renumbering the simulations paths after each time step, which allows a lower dimensional quasirandom sequence of points to be used.

1 Introduction

Diffusion Monte Carlo (DMC) [5, 9, 27, 31] is a common method for estimating the properties of quantum mechanical systems by computing averages over sets of random walk simulations. It is closely related to Green's Function Monte Carlo (GFMC) [4, 14, 17].

Quasirandom sequences [10, 24, 26] are sequences of points designed to produce more accurate integration estimates, called quasi-Monte Carlo, than standard Monte Carlo with random or pseudorandom points. This is accomplished by relaxing the requirement that successive points be independent, and instead seeking a set of points which spreads out over the domain as uniformly as possible. In one dimension, the best such sequence is a uniform grid, but in higher dimensions more complicated sequences are preferable [7, 24]. The particular quasirandom sequence which we have used is called the base-2 Niederreiter sequence [3].

*At Westinghouse Bettis Atomic Power Laboratory, West Mifflin, PA, as of 11/94. This work was done while at UCLA. Research supported in part by the Air Force Office of Scientific Research under grant number F49620-94-1-0091.

Although originally used for integration estimates, quasirandom sequences can be used for simulations as well. There have been several examples of such quasi-Monte Carlo simulations [11, 16, 25, 28], however they have often been limited to cases in which the number of time steps or iterations was small.

We have found that for Diffusion Monte Carlo, which often involves hundreds or thousands of time steps, we can use quasirandom sequences by making use of a recently developed technique involving renumbering the simulations after each time step [20, 22]. Without this technique either the dimensions of the quasirandom sequence would need to be too high to be effective (quasirandom sequences lose their effectiveness as the dimensions increase [2, 18, 21, 23, 26]), or else correlation errors between time steps would lead to erroneous results. With quasirandom sequences and renumbering, Diffusion Monte Carlo results can be significantly improved in terms of expected error versus either number of simulations or total cpu time, as will be demonstrated in the examples below.

Note: The renumbering technique mentioned above is still in its infancy and has not yet been extensively tested nor has its convergence been proven in all cases. However, computational results using this technique so far have been strongly encouraging [22]. The results presented here represent a further confirmation that renumbering can be effective, and that it has a potentially wide variety of useful applications.

2 Diffusion Monte Carlo

In this section a brief description of Diffusion Monte Carlo will be given. As mentioned above, Diffusion Monte Carlo is a method for estimating the properties of quantum mechanical systems. For simplicity, the property which we shall deal with will be the ground state energy. Diffusion Monte Carlo involves converting the time-dependent Schrödinger Equation into a diffusion equation by changing the time variable to imaginary time, as shown below. The equation is then multiplied by a 'trial function,' which is a rough estimate of the exact solution, in order to reduce the expected errors. (Note: This is a type of variance reduction technique [9].) Monte Carlo estimates are computed by following a set of random 'walkers' as they drift and diffuse through the system according to the diffusion equation above and then averaging their trial energy levels.

2.1 Schrödinger Equation

For a quantum mechanical system of M bodies with a time-independent potential function, the time-dependent Schrödinger Equation is the following:

$$-\sum_{k=1}^{M} \frac{\hbar^2}{2m_k} \nabla_k^2 \psi(\boldsymbol{y},t) + v(\boldsymbol{y})\,\psi(\boldsymbol{y},t) = i\hbar\,\frac{\partial}{\partial t}\psi(\boldsymbol{y},t) \tag{1}$$

where m_k is the mass of the k'th body, $\boldsymbol{y} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M)$ is a $3M$-dimensional vector representing the position of all of the bodies, ∇_k^2 is the Laplacian operator for the k'th body, $v(\boldsymbol{y})$ is the potential function, and $\psi(\boldsymbol{y},t)$ is the wavefunction or probability amplitude for the system.

Defining the Hamiltonian operator, $H = -\sum_{k=1}^{M} \frac{\hbar^2}{2m_k}\nabla_k^2 + v(\boldsymbol{y})$, (1) can be rewritten,

$$H\psi(\boldsymbol{y},t) = i\hbar\frac{\partial}{\partial t}\psi(\boldsymbol{y},t)$$

A formal solution to (1) is the following:

$$\psi(\boldsymbol{y},t) = \sum_{j=0}^{\infty} a_j\, e^{-i\hbar^{-1}E_j t}\psi_j(\boldsymbol{y})$$

when v is continuous, $\psi_j(\boldsymbol{y})$, $j = 0,\ldots$, are a complete orthogonal set of eigenfunctions of H with $E_0 < E_1 < \cdots$, and a_j is the $\psi_j(\boldsymbol{y})$ component of $\psi(\boldsymbol{y},0)$, $a_j = \int_{\Re^{3M}} \psi(\boldsymbol{y},0)\,\psi_j(\boldsymbol{y})\,d\boldsymbol{y}$.

To transform to imaginary time, let $\phi(\boldsymbol{y},\tau) = \psi(\boldsymbol{y},it)$, then

$$H\phi(\boldsymbol{y},\tau) = -\,\hbar\,\frac{\partial}{\partial\tau}\phi(\boldsymbol{y},\tau) \tag{2}$$

This has formal solution,

$$\phi(\boldsymbol{y},\tau) = \sum_{i=0}^{\infty} a_i\, e^{-\hbar^{-1}E_i\tau}\psi_i(\boldsymbol{y})$$

where the eigenfunctions and eigenvalues are unchanged since H is a purely spatial operator. (2) is a diffusion equation which, unlike (1), converges exponentially fast to the ground state, ψ_0. The only exception is when the wavefunction is specifically restricted to be orthogonal to ψ_0, as when higher energy states are being sought.

2.2 Trial Function

A trial function, which is a rough estimate of the exact solution, is multiplied into the diffusion equation, (2), in order to reduce the variance and eliminate discontinuities.

Rewriting (2) as follows:

$$\frac{\partial}{\partial\tau}\phi(\boldsymbol{y},\tau) = \sum_{k=1}^{M}\frac{\hbar}{2m_k}\,\nabla_k^2\phi(\boldsymbol{y},\tau) - \frac{1}{\hbar}\,v(\boldsymbol{y})\,\phi(\boldsymbol{y},\tau) \tag{3}$$

we see a partial differential equation corresponding to diffusion and growth.

The potential, $v(\boldsymbol{y})$, acts in the growth term. When it is highly variable or singular (for example a Coulomb potential) it leads to high variance Monte Carlo estimates. Multiplying by a well-chosen trial function, ψ_T, can eliminate discontinuities and reduce the variance of the Monte Carlo estimates.

Let $f(\boldsymbol{y},\tau) = \psi_T(\boldsymbol{y})\,\phi(\boldsymbol{y},\tau)$, then multiplying (3) by ψ_T and rearranging terms gives us the following:

$$\frac{\partial}{\partial\tau}f(\boldsymbol{y},\tau) = \sum_{k=1}^{M}\frac{\hbar}{2m_k}\,\nabla_k^2 f(\boldsymbol{y},\tau) - \sum_{k=1}^{M}\nabla_k\cdot[\boldsymbol{a}_k(\boldsymbol{y})\,f(\boldsymbol{y},\tau)] - b(\boldsymbol{y})\,f(\boldsymbol{y},\tau) \tag{4}$$

where

$$b(\boldsymbol{y}) = \frac{H\psi_T(\boldsymbol{y})}{\psi_T(\boldsymbol{y})} \text{ , the trial energy function.}$$

$$\boldsymbol{a}_k(\boldsymbol{y}) = \frac{\hbar}{m_k} \cdot \frac{\nabla_k \psi_T(\boldsymbol{y})}{\psi_T(\boldsymbol{y})} \text{ , } k = 1, \ldots, M \text{ , the drift functions.}$$

The new equation (4) is similar to (3) but with an extra set of terms with 1st derivatives of f, which add a drift to the diffusion process. Importantly, b, the trial energy, replaces v, the potential, in the growth term. When ψ_T has been well chosen, the trial energy will be relatively constant, leading to lower variance Monte Carlo estimates of E_0. Note: Hypothetically, if one used exactly $\psi_T = \psi_0$ then b would be a constant, equal to E_0, and trivial zero-variance Monte Carlo estimates would result.

In practice the ground state energy, E_0, is typically estimated using what is called the variational estimate. Let $\psi_*(\boldsymbol{y})$ be an arbitrary wavefunction with a non-zero ground state component. Then using orthogonality properties,

$$E_0 = \frac{\int_{\Re^{3M}} H\psi_*(\boldsymbol{y})\, \psi_0(\boldsymbol{y})\, d\boldsymbol{y}}{\int_{\Re^{3M}} \psi_*(\boldsymbol{y})\, \psi_0(\boldsymbol{y})\, d\boldsymbol{y}}$$

This is estimated by computing,

$$E(\tau) = \frac{\int_{\Re^{3M}} H\psi_*(\boldsymbol{y})\, \phi(\boldsymbol{y}, \tau)\, d\boldsymbol{y}}{\int_{\Re^{3M}} \psi_*(\boldsymbol{y})\, \phi(\boldsymbol{y}, \tau)\, d\boldsymbol{y}} \tag{5}$$

and then letting $E(\tau) \to E_0$ as $\tau \to \infty$ and $\phi(\boldsymbol{y}, \tau) \to \psi_0(\boldsymbol{y})$.

It is natural then to take $\psi_* = \psi_T$. Then we have,

$$E(\tau) = \frac{\int_{\Re^{3M}} H\psi_T(\boldsymbol{y})\, \phi(\boldsymbol{y}, \tau)\, d\boldsymbol{y}}{\int_{\Re^{3M}} \psi_T(\boldsymbol{y})\, \phi(\boldsymbol{y}, \tau)\, d\boldsymbol{y}} = \frac{\int_{\Re^{3M}} b(\boldsymbol{y})\, f(\boldsymbol{y}, \tau)\, d\boldsymbol{y}}{\int_{\Re^{3M}} f(\boldsymbol{y}, \tau)\, d\boldsymbol{y}} \tag{6}$$

In which case we observe that $E(\tau)$ can be computed as the expected value of $b(\boldsymbol{y})$ when $\boldsymbol{y}$ is distributed according to $f(\boldsymbol{y}, \tau)$ as a probability density.

Diffusion Monte Carlo consists of estimating $E(\tau)$ by using the diffusion, drift, and growth rates from Equation 4 to generate a large set of simulation paths, or random walkers, distributed in space according to $f(\boldsymbol{y}, \tau)$, and then averaging b over the set of paths to compute a Monte Carlo estimate of $E(\tau)$. As long as the simulations have traveled for a long enough imaginary time, τ, so that $E(\tau)$ has converged to E_0, we have our desired estimate of E_0.

One important assumption above is that $f(\boldsymbol{y}, \tau)$ is strictly positive, so that it can function as a probability density. In many systems with fermions, however, the wavefunction must be negative in certain regions, leading f to be negative as well, unless one knows beforehand exactly where the negative regions will be. As a result, further approximations such as the fixed node approximation [4] are necessary. For simplicity, we will only consider examples in which f is strictly positive so this problem does not arise. For more details on this active area of research see [1, 4, 32].

3 Stochastic Simulation

In this section, a few of the details of simulating the random walks for Diffusion Monte Carlo will be discussed. The stochastic process which corresponds to (4) in the previous section is governed by the following system of stochastic differential equations:

$$\begin{aligned} dZ_k(\tau) &= a_k(Z(\tau))\,d\tau + \sigma_k\,dW \;\;,\;\; k=1,\ldots,M \\ dY(\tau) &= -b(Z(\tau))\,d\tau \end{aligned}$$

where $Z=(Z_1,\ldots,Z_M)$, $Z(0)\sim f(\boldsymbol{y},0)$ (initial distribution), $Y(0)=0$, $\sigma_k=\sqrt{\hbar/m_k}$.

For a random walker, $Z(\tau)$ indicates the position in $3M$-dimensional space and $Y(\tau)$ indicates the natural log of the growth weight associated with that walker as it travels along its path. (Note: Sometimes walkers are allowed to split into multiple walks or terminate so that weights are unnecessary.)

Using Ito's Formula, one can prove the following important result:

$$E(\tau)=\frac{E\left[b(Z(\tau))e^{Y(\tau)}\right]}{E\left[e^{Y(\tau)}\right]} \;\;,\;\; \text{over stochastic paths } (Z,Y). \tag{7}$$

where the quantities on the right hand side are expectations over the set of all possible stochastic paths followed by a random walker.

In order to generate simulation paths for $(Z(\tau),Y(\tau))$ it is necessary to discretize the imaginary time. This is commonly done using the *Euler-Maruyama* scheme:

$$\begin{aligned} Z_k^{(j+1)} &= Z_k^{(j)} + \sqrt{h}\,\sigma_k W_k^{(j)} + h\,a_k(Z^{(j)}) \;\;,\;\; k=1,\ldots,M \\ Y^{(j+1)} &= Y^{(j)} - h\,b(Z^{(j)}) \end{aligned}$$

where $Z^{(0)}\sim\psi_T^2(\boldsymbol{y})$ to approximate $\psi_T(\boldsymbol{y})\psi_0(\boldsymbol{y})$, $Y^{(0)}=0$, h is the time step size and j the time step counter so that $\tau=hj$, and $W_k^{(j)}$ is a $3D$ vector of Gaussian, $\mathcal{N}(0,1)$, random variables. For estimates of $E(\tau)$ as an expected value, this scheme is first order accurate in h (Kloeden and Platen [15]).

In order to take larger time steps, without large discretization errors, one can use higher order discretization schemes. In particular, the following scheme is third order accurate (for estimates of $E(\tau)$). It is related to Runge-Kutta methods for ODE's and was derived by Helfand and Greenside ([8, 12]):

$$\begin{aligned} g_{1,k} &= a_k(Z^{(j)}+\sqrt{h}\,\zeta_1) \;\;,\;\; k=1,\ldots,M \\ g_{2,k} &= a_k(Z^{(j)}+h\,.516719\,g_1+\sqrt{h}\,\zeta_2) \\ g_{3,k} &= a_k(Z^{(j)}+h(-.397300\,g_1+.427690\,g_2)+\sqrt{h}\,\zeta_3) \\ g_{4,k} &= a_k(Z^{(j)}+h(-1.587731\,g_1+1.417263\,g_2+1.170469\,g_3)+\sqrt{h}\,\zeta_4) \\ Z^{(j+1)} &= Z^{(j)}+h(.644468\,g_2+.194450\,g_3+.161082\,g_4)+\sqrt{h}\,\zeta_0 \\ Y^{(j+1)} &= Y^{(j)}-.5\,h\left(b(Z^{(j)})+b(Z^{(j+1)})\right) \end{aligned}$$

where $Z^{(0)}\sim\psi_T^2(\boldsymbol{y})$, $Y^{(0)}=0$, $\zeta_0=W_k^{(j)}$, $\zeta_1=.271608\,V_k^{(j)}$, $\zeta_2=.516719\,W_k^{(j)}+.499720\,V_k^{(j)}$, $\zeta_3=.030390\,W_k^{(j)}-.171658\,V_k^{(j)}$, $\zeta_4=V_k^{(j)}$, and $W_k^{(j)}$ and $V_k^{(j)}$ are two independent $3D$ vectors of Gaussian, $\mathcal{N}(0,1)$, random variables. This scheme involves more

computation than Euler-Maruyama and twice the number of Gaussian random variables per time step. However, convergence in imaginary time, τ, is reached in far fewer time steps by taking much larger time steps.

Note: To be precise, the weights, Y, are only simulated to second order accuracy in terms of h above. In addition, as discussed in [6], one should be careful using a higher order accurate scheme because of possible hidden errors. In the examples discussed below, any such errors were found to be below the level of Monte Carlo errors and were therefore not considered significant.

4 Standard Pseudorandom Diffusion Monte Carlo

Standard pseudorandom Diffusion Monte Carlo consists of using the simulation techniques of the last section to compute an estimate of E_0 using (7), as follows:

$$\hat{E}_N = \frac{\sum_{i=1}^{N} b(Z_i^{(m)})\, e^{Y_i^{(m)}}}{\sum_{i=1}^{N} e^{Y_i^{(m)}}} \tag{8}$$

where $(Z_i^{(m)}, Y_i^{(m)})$ represents the position and weight of the i'th stochastic simulation path at time $\tau = mh$. Simulation paths are computed using pseudorandom numbers which are transformed into Gaussian random variables using any one of a number of standard techniques [19].

The error in this estimate can be broken down as follows:

$$\begin{aligned} \hat{E}_N - E_0 &= [\hat{E}_N - E^{(h)}(m)] + [E^{(h)}(m) - E(\tau)] + [E(\tau) - E_0] \\ &= \epsilon_{mc} + \epsilon_h + \epsilon_\tau \\ &= O(N^{-1/2}) + O(h^\alpha) + O(e^{-\tau}) \end{aligned}$$

where $E^{(h)}(m)$ is defined as the expected value of $\hat{E}_N$ as $N \to \infty$, and α is the order of accuracy of the time discretization method.

In the equation above, ϵ_{mc} is the statistical or 'Monte Carlo' error, ϵ_h is the approximation error from the time discretization, and ϵ_τ is the convergence in imaginary time error.

4.1 Ensemble-Based Sampling

In practice, the estimate (8) above is seldom used precisely as written. Instead of generating N independent stochastic simulation paths through time τ, it is much more common to follow a smaller set, or ensemble, of R 'random walkers' as they travel forward in imaginary time. After a start-up period of J time steps, which reduces the time convergence error, ϵ_τ, to acceptably small levels, the positions and weights of the walkers are used after every j'th time step in computing the estimate of E_0, where j is taken sufficiently large so that the new positions are (nearly) independent of the previous positions used. For example, for a total sample size of $N = 10000$, an ensemble size of 50 walkers would

be followed for $J + 200j$ time steps in order to accumulate the full sample. J and j are determined empirically.

Ensemble-based sampling saves memory, since R can be much smaller than N, and time, since j can be much smaller than J. As long as h is sufficiently small so that ϵ_h is negligible, and J is sufficiently large so that ϵ_τ is negligible, the primary source of error left is the Monte Carlo error, ϵ_{mc}.

5 Quasirandom Diffusion Monte Carlo

In this section we examine how quasirandom sequences can be used to reduce the Monte Carlo error for Diffusion Monte Carlo estimates.

Quasirandom sequences are designed to produce more accurate integration estimates than standard Monte Carlo when used in place of random or pseudorandom points. By spreading out as uniformly as possible over the integration domain, quasirandom sequences can produce Monte Carlo errors which converge at rates as fast as $O(N^{-1})$ instead of the usual $O(N^{-1/2})$ for Monte Carlo [24].

However, quasirandom Monte Carlo estimates, called quasi-Monte Carlo, lose their advantage over standard Monte Carlo as the number of dimensions increases. This is a well established result, theoretically and computationally [2, 18, 21, 23, 26]. While there is no specific maximum number of dimensions for which quasi-Monte Carlo is useful, it appears that for many applications once the number of dimension reaches levels near 30 or 40 quasi-Monte Carlo may not be any better than standard pseudorandom Monte Carlo, and at higher dimensions quasi-Monte Carlo can often produce results which are worse than standard Monte Carlo. This limitation on the number of dimensions has important implications for quasirandom Diffusion Monte Carlo, as we shall see below.

5.1 Quasirandom Simulation

Although originally designed for integration estimates, quasirandom sequences have been used for simulations as well. This is accomplished by treating a simulation problem as being equivalent to the evaluation of an integral of the expected output over the space of all possible simulation paths [22, 29]. The integrand is then a function of all the random inputs at every time step of the simulation path since ultimately the output is a function of all of these inputs. Therefore, the total number of dimensions is equal to the sum of the dimensions, or number of inputs, at each time step. For a simulation path consisting of m time steps in d-dimensions then, we would have md dimensions. Hence, the number of dimensions can easily become very high whenever many time steps are involved, as is typically the case for Diffusion Monte Carlo. As discussed above, this means that quasi-Monte Carlo may not be effective in such instances since it loses its effectiveness as the number of dimensions increases. As a result, most previous applications of quasi-Monte Carlo estimates to simulations have been limited primarily to simulations involving only a few time steps or iterations.

Recently a way to overcome the limitation on time steps has been found which involves

renumbering, or scrambling, the order of a set of simulation paths after each time step [16, 20, 22]. As discussed in [22], this technique can effectively reduce the number of dimensions from md to simply d (or $d+1$) by separating each time step apart from those preceding and following it.

5.2 Application to Diffusion Monte Carlo

For Diffusion Monte Carlo, the number of random input variables for each time step is equal to $3M$, where M is the number of bodies in the quantum mechanical system, for the Euler-Maruyama scheme. For the higher order Helfand-Greenside scheme this doubles to $6M$ per time step. Therefore, regardless of any renumbering, there is an essential limitation to the size of the quantum mechanical systems for which quasirandom Diffusion Monte Carlo may be useful. The examples discussed below include a three dimensional harmonic oscillator ($M = 1$) and the Helium atom ($M = 2$, fixed nucleus). It is anticipated that similar results should be obtainable in the near future for systems of 3 or 4 bodies as well. For larger systems of say 10 or more bodies, it is not anticipated that quasirandom Diffusion Monte Carlo, as presented here, will be useful.

The use of renumbering is essential since Diffusion Monte Carlo typically involves hundreds or thousands of time steps, m, which without renumbering would lead to dimensions, $3Mm$ or $6Mm$, far too high for quasi-Monte Carlo to be effective. Renumbering allows us to use just $3M$ or $6M$ dimensional quasirandom points regardless of the number of time steps, as long as renumbering is done following each time step. This makes quasirandom Diffusion Monte Carlo feasible.

5.3 Renumbering

Renumbering, described in detail in [22], involves generating a full set of simulation paths together, and, after each time step, scrambling up the order of the numbering of the simulation paths. The method of renumbering can be either random – using a random permutation of $\{1, \ldots, N\}$ to renumber N paths, or a more sophisticated technique known as binary renumbering, in which paths are renumbered according to where they are located in space after each step.

For binary renumbering, which is used in the examples below, after each time step, simulation paths are grouped according to which subregion of space they each fall within. Subregions are formed by dividing each dimension into L levels where L is a power of two (for convenience), creating a total of L^d subregions in d-dimensions. When every path's subregion has been determined, the paths are renumbered by taking all of the paths in one subregion followed by all the points in the next subregion, and so on through all the subregions. The order within each subregion is simply random. The order of the subregions themselves is fixed in a way that minimizes the distance between successive subregions. This particular arrangement is used because it reduces the average distance between successively numbered paths, which (as discussed in [22]) leads to improved quasi-Monte Carlo results by reducing a quantity known as the variation.

It is important to mention at this point that without renumbering, if we used just $3M$ or $6M$ dimensional quasirandom points for Diffusion Monte Carlo, correlations between the quasirandom points used in successive time steps would lead to biased and incorrect results. The other alternative is to use extremely high dimensional quasirandom points, which as discussed previously, is not recommended because quasirandom points tend to lose their effectiveness as the number of dimensions increases.

5.4 Continuation Method

Given the very high number of time steps required in general for Diffusion Monte Carlo, since τ must be large enough so that the walkers have converged in imaginary time to the ground state (so that e_τ is sufficiently small), while h must be small enough so that discretization error, e_h, is sufficiently small as well, it is impractical to use quasirandom sequences throughout the simulations. Renumbering allows us to use $3M$ or $6M$ dimensional quasirandom points regardless of the number of time steps, but going through the renumbering process after every one of a large number of time steps would add a great deal of extra work, which it would be preferable to avoid. A further difficulty is ensemble-based sampling, described in Section 4.1, to which there is no clear way to apply quasirandom sequences effectively.

A solution to the two difficulties posed above is to combine standard pseudorandom ensemble-based sampling with quasirandom sampling in what is termed a 'continuation' method. For continuation, we first use standard Monte Carlo with ensemble-based sampling to generate a set of N pseudorandom simulation path 'endpoints' and weights, $\{Z_i, Y_i\}_{i=1}^N$. (Note: Endpoints is in quotations because with ensemble-based sampling, the positions and weights used are actually intermediate values for a smaller set of random walkers.) Then, quasirandom points are used to 'continue' the paths from each (Z_i, Y_i) for an additional fixed number of time steps, using renumbering after each of these steps, after which a Monte Carlo estimate of E_0 is computed as the weighted average of b at the set of new 'continued' endpoints. (Note: For standard Monte Carlo, the estimate would be computed as a weighted average of b at the original 'non-continued' endpoints.)

The basic idea motivating this procedure is that we are first generating a sample, $\{Z_i, Y_i\}_{i=1}^N$, which is relatively free of errors ϵ_h and ϵ_τ, and dominated by error ϵ_{mc}. Then, the additional quasirandom steps are used to reduce this Monte Carlo error from $O(N^{-1/2})$ levels to accelerated levels closer to $O(N^{-1})$, which are characteristic of quasi-Monte Carlo.

One drawback of continuation is that by requiring us to store the entire set of N pseudorandom endpoints before continuing them, this technique negates the memory savings of ensemble-based sampling. This could be a significant drawback when N is large. The same problem arises, however, if an entirely quasirandom simulation is done without ensemble-based sampling and without continuation. Variations in which the samples are continued in smaller blocks to reduce the memory requirements might be possible [30].

On the other hand, continuation can produce a large improvement in cpu time, since the number of renumberings can be limited to just the number of continuation steps rather than the total number of time steps. Then, as long as the number of quasiran-

dom continuation steps is relatively small, the additional cpu time required will not be excessive.

Since increasing the number of continuation steps allows the continued paths to travel further away from the original pseudorandom endpoints, resulting in a better opportunity for Monte Carlo errors to be significantly reduced, but also costing extra cpu time, there is a trade-off between decreased error and increased cpu time as the number of continuation steps increases. In practice, the optimal number of continuation steps can be determined empirically by observing the resultant estimated errors and average cpu times.

It is at this point that using a higher order accurate discretization method can be beneficial, since it allows larger time steps to be used, With larger time steps, fewer continuation steps and renumberings are needed in order to move sufficiently far away from the original pseudorandom endpoints to see an appreciable improvement.

The examples below will demonstrate the effectiveness of quasirandom Diffusion Monte Carlo using the continuation method with renumbering.

6 Quantum Mechanical Examples

We examine two quantum mechanical examples to study the effectiveness of quasirandom Diffusion Monte Carlo using continuation and renumbering, as discussed above, when compared with standard pseudorandom Diffusion Monte Carlo. In all cases, the high-order accurate Helfand-Greenside discretization scheme is used. This allows larger time steps to be taken than if a less accurate method, such as Euler-Maruyama, were used instead. The two examples are a three dimensional harmonic oscillator and the Helium atom.

6.1 Error Measurement

With quasi-Monte Carlo estimates, the issue of error measurement is always an important one since no truly satisfactory standard procedure has yet been established. Let N be the number of simulations or equivalently the 'sample size.' Then, for each of the examples, each different estimate of E_0, pseudorandom or quasirandom, is repeatedly computed R times at each of several different fixed values of N. This allows the expected error as a function of sample size to be observed.

Ideally, each repetition should be independent. True independence is rare when using deterministic sequences of any kind, including the best pseudorandom generators. However, when using quasirandom sequences this is a particular concern because successive points can be highly correlated. In the present examples, each set of R quasirandom estimates for a given sample size, N, is made using R successive blocks of points in the quasirandom sequence (the base-2 Niederreiter sequence [3]). However, in order to try to ensure that these estimates are not too highly correlated, a random number of points between 1 and N is skipped between each block of points.

It is not inconceivable that given just the right scenario, successive quasirandom estimates could be correlated, thereby skewing the error estimates made below, particularly

the second one when the exact solution is unknown. Although no evidence of any such problem was found in the examples below, in which the exact solutions, or highly accurate estimates, were known in advance, the problem of reliable error estimation remains a critical area for further research in quasi-Monte Carlo because without adequate error estimation techniques these methods will not be widely adopted for use.

Assuming the independence discussed above, the root mean square error of each estimate of E_0 at sample size N can itself be estimated as follows:

$$\hat{e}_{rms}(N) = \sqrt{\frac{1}{R}\sum_{r=1}^{R}(\hat{E}_N^{(r)} - E_0)^2}$$

where $\hat{E}_N^{(r)}$ is the r'th estimate at sample size N and E_0 is the exact ground state energy.

An estimate of the error which does not rely on the exact solution (presumably unknown in any practical situation) is also needed. For non-biased estimates, this can be accomplished by computing the sample variance of each set of R estimates computed and then taking the square root as follows:

$$\hat{\sigma}(N) = \sqrt{\frac{1}{R-1}\sum_{r=1}^{R}(\hat{E}_N^{(r)} - \bar{E})^2}$$

where $\bar{E}$ is the average of all R estimates. In the examples below, both estimates, $\hat{e}_{rms}(N)$ and $\hat{\sigma}(N)$ were computed, with similar results in either case.

6.2 Harmonic Oscillator

This is a simple example which readily demonstrates the effectiveness of the quasirandom techniques.

Example 1 *The ground state energy, E_0, is estimated for a three dimensional harmonic oscillator. The potential energy function is the following:*

$$v(x,y,z) = \frac{1}{2}k\,r^2$$

where $k > 0$ is the oscillator strength, and $r^2 = x^2 + y^2 + z^2$ is the squared distance from the oscillator's center, which is taken to be the origin.

In atomic units, with $\hbar = 1$, we consider a harmonic oscillator of mass $m = 1$ and strength $k = 1$, which has an exact ground state energy of $E_0 = 1.5$. This exact quantity is used in computations of the estimated root mean square error, $\hat{e}_{rms}(N)$. Note: In atomic units, energy is measured in units of $4.36 \cdot 10^{-18}$ Joules (Hartrees), length in units of $5.3 \cdot 10^{-11}$ meters (Bohrs), and time in units of $2.4 \cdot 10^{-17}$ seconds.

The exact ground state wave function, $\psi_0(x,y,z)$, is equal to $(4\pi)^{-1}e^{-r^2/2}$, leading us to use as our trial function (as described in Section 2.2):

$$\psi_T(x,y,z) = (3.92157\pi)^{-1}e^{-.51r^2}$$

where we deliberately take a function slightly different from the exact ground state in order to avoid the trivial case of exact zero-variance results.

The resultant trial energy function is then the following:

$$b(\boldsymbol{y}) = \frac{H\psi_T(\boldsymbol{y})}{\psi_T(\boldsymbol{y})} = 1.53 - 0.202r^2$$

While the vector drift function is the following:

$$\boldsymbol{a}(\boldsymbol{y}) = (-1.02x, -1.02y, -1.02z)$$

For the simulations, the time steps are of size $h = 0.15$ using the Helfand-Greenside discretization scheme (see Section 3). This time step size was determined to be small enough so that discretization errors were insignificant relative to Monte Carlo errors. An ensemble size of $R = 50$ random walkers is used, with a delay of $J = 70$ time steps before positions are stored, and a gap of $j = 12$ time steps between successively stored positions (see Section 5.4).

Standard pseudorandom Diffusion Monte Carlo is compared with quasirandom Diffusion Monte Carlo using continuation (with 2, 4, and 12 continuation steps) and renumbering (with each dimension divided into 64 equally spaced levels between -5 and +5 for a total of 262144 subregions in three dimensions).

Figure 1 is a plot of the error, $\hat{e}_{rms}(N)$, computed over $R = 60$ repetitions, as a function of N on a log-log scale. The log-log scale allows the plot of an error of the form $e = b\,N^a$ to appear as a straight line with slope a. We expect a convergence rate of $a = -0.5$ for standard Monte Carlo, and a lower (or faster) rate for quasi-Monte Carlo.

Figure 2 is a similar plot of the error against average cpu time instead of N. The graph is intended to provide a more realistic comparison of the various estimates of E_0.

6.2.1 Discussion of Results: Example 1

We see in Figures 1 and 2 that quasirandom Diffusion Monte Carlo clearly outperforms standard Diffusion Monte Carlo for this example. The convergence rate of the error improves from close to $O(N^{-1/2})$ for the standard method to $O(N^{-0.641})$ for the quasirandom method, with this rate steadily improving as the number of continuation steps is increased.

Figure 2 makes it clear that the extra cpu time required for renumbering and continuation is more than made up for by the reduced errors. For an error level of about 10^{-4}, we see a reduction in cpu time by a factor of about 5 for quasirandom Diffusion Monte Carlo with either 4 or 12 continuation steps.

We also observe that taking any more than 12 continuation steps would probably not be worthwhile since in Figure 2 the results using 12 steps, in terms of cpu time, are comparable to the results using just 4 steps.

6.3 Helium Atom

For the Helium atom, a number of additional features are incorporated into the quasirandom simulations. First of all, in order to reduce storage requirements by nearly half,

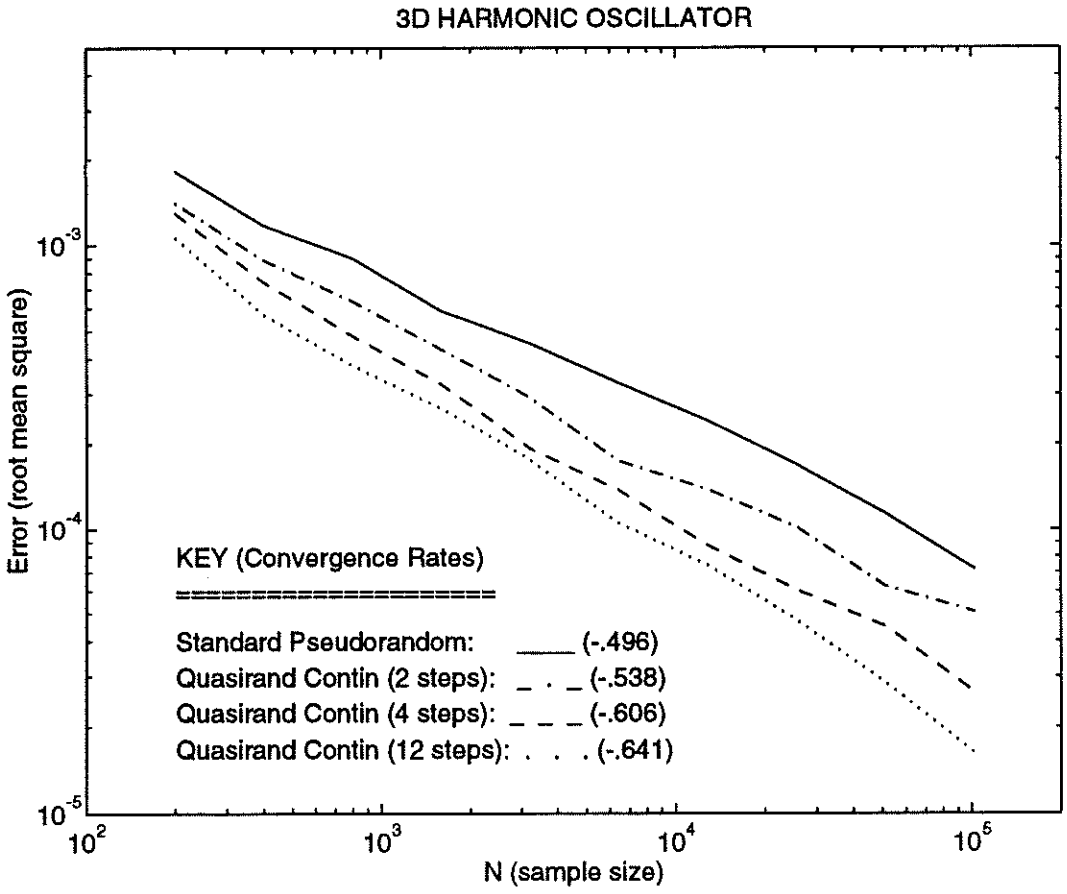

Figure 1: Log-Log Plot for Example 1.

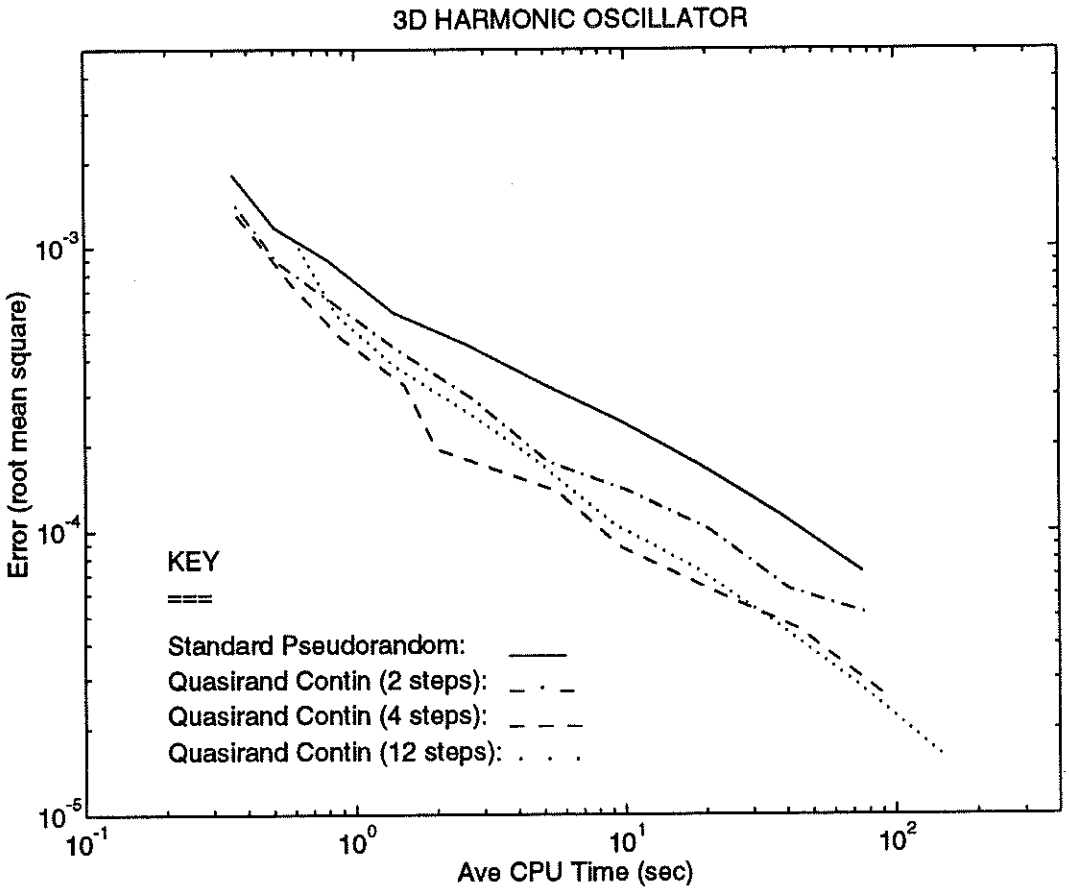

Figure 2: Log-Log Plot (Timing) for Example 1.

some symmetries are exploited. Given configuration, $\boldsymbol{y} = (x_1, y_1, z_1, x_2, y_2, z_2)$, the potential function and therefore the wavefunction solution is only dependent on r_1, r_2, and r_{12}, the distance from the first electron to the nucleus, the second electron to the nucleus, and between the electrons, respectively. Therefore, when the pseudorandom endpoints are stored to be reused for continuation, only the triplet r_1, r_2, r_{12} is saved rather than all six dimensions. The same thing is done for each of the continuation steps as well.

During the continuation steps, for each path, the missing angles are sampled quasirandomly at each time step after retrieving the three distances from storage and before taking the next time step. This reduces the storage required by nearly a half, and it also, importantly, simplifies the renumbering process because instead of requiring subregions over $6D$ space, subregions over only $3D$ space are necessary. This use of symmetry does not introduce any new approximation errors into the results, and should be applicable to many quantum mechanical systems with various symmetries.

A second feature of the Helium atom simulations is that in the renumbering process, an adaptive method is used to select the location of subregions. Initially the $3D$ space is divided into 4096 subregions by dividing each dimension into 16 levels. Then, depending on where most of the pseudorandom endpoints fall, some of these subregions are further subdivided into 512 smaller subregions (8 more sublevels in each dimension). This method gives us fine subdivisions in regions of greater interest where more of the simulation paths are, and coarser subdivisions in lower interest regions where fewer simulation paths appear.

Example 2 *The ground state energy, E_0, of the Helium atom is estimated using Diffusion Monte Carlo. Experimental results [6] have produced the following estimated value: $E_O = -2.903724$, in atomic units, which shall be considered the 'exact' solution for the purpose of computing errors here. We assume a fixed nucleus located at the origin so that the system is just a two body system ($M = 2$).*

The potential energy function, in atomic units, is the following:

$$v(\boldsymbol{y}) = -\frac{2}{r_1} - \frac{2}{r_2} + \frac{1}{r_{12}}$$

where $\boldsymbol{y} = (x_1, y_1, z_1, x_2, y_2, z_2)$, r_1 is the distance from the first electron to the nucleus, r_2 the distance from the second electron to the nucleus, and r_{12} the distance between the electrons.

The trial function chosen is of the form:

$$\psi_T(\boldsymbol{y}) = \exp(-2r_1)\exp(-2r_2)\exp\left(\frac{a\, r_{12}}{1 + b\, r_{12}}\right)$$

with $a = 0.5$ and $b = 0.2$, which ensures that trial energy, $b(\boldsymbol{y})$, remains finite at $r_1 = 0$, $r_2 = 0$, and $r_{12} = 0$ [13]. The resultant trial energy function is then the following:

$$b(\boldsymbol{y}) = -4 + \frac{1}{r_{12}} - \frac{1}{r_{12}(1 + b\, r_{12})^3} - \frac{1}{4(1 + b\, r_{12})^4} + \frac{(r_1 + r_2)(1 - r_1 \cdot r_2/r_1 r_2)}{r_{12}(1 + b\, r_{12})^2}$$

While the vector drift function is the following:

$$\boldsymbol{a}(\boldsymbol{y}) = \frac{\boldsymbol{y} - \boldsymbol{y}^*}{2r_{12}(1 + b\, r_{12})^2} - \frac{2\boldsymbol{y}}{r_1}$$

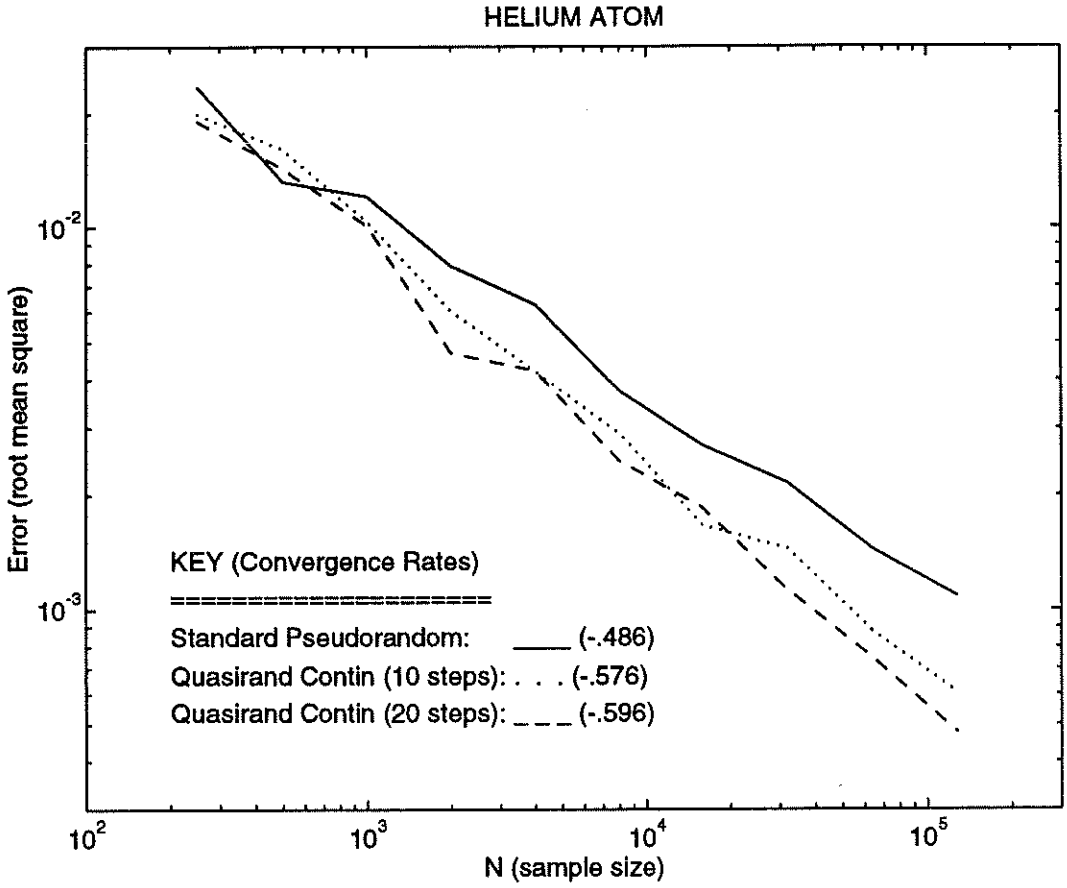

Figure 3: Log-Log Plot for Example 2.

where $y^* = (x_2, y_2, z_2, x_1, y_1, z_1)$.

For the simulations, the time steps are of size $h = 0.03$ *and the Helfand-Greenside discretization scheme is used. As in Example 1, the time step size was chosen to be small enough so that discretization errors were insignificant relative to Monte Carlo errors. An ensemble size of* $R = 100$ *random walkers is used, with a delay of* $J = 500$ *time steps before positions are stored, and a gap of* $j = 50$ *time steps between successively stored positions.*

Standard pseudorandom Diffusion Monte Carlo is compared with quasirandom Diffusion Monte Carlo using continuation (with 10 and 20 continuation steps) and renumbering (using the adaptive technique described above).

Figure 3 is a plot of the error, $e_{rms}(N)$*, computed over* $R = 60$ *repetitions, as a function of* N *on a log-log scale. As in Example 1, the log-log scale allows the plot of an error of the form* $e = b\,N^a$ *to appear as a straight line with slope* a*. We expect* a *to be near* -0.5 *for standard Monte Carlo and lower for quasi-Monte Carlo. Figure 4 is a similar plot of the error against average cpu time instead of* N*.*

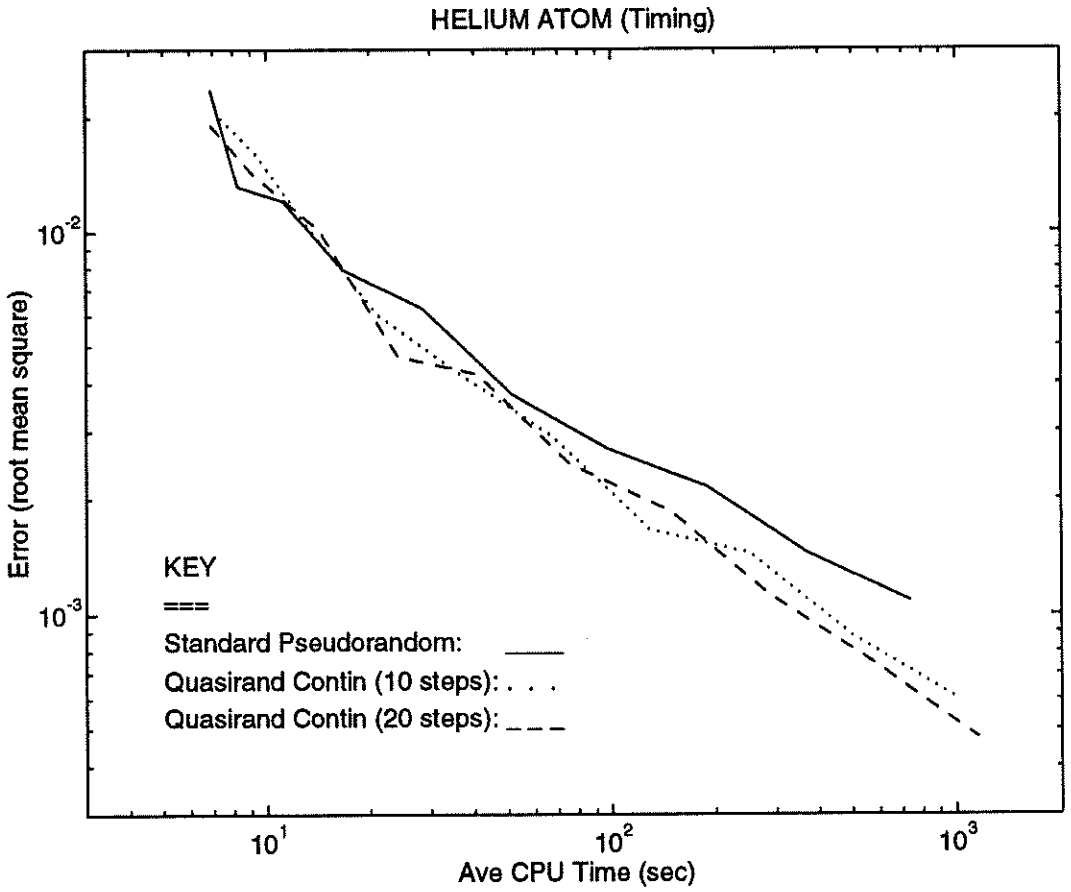

Figure 4: Log-Log Plot (Timing) for Example 2.

6.3.1 Discussion of Results: Example 2

The results for Example 2 are similar to those for the first example. We see in Figures 3 and 4 that quasirandom Diffusion Monte Carlo still clearly outperforms standard Diffusion Monte Carlo, although not to as large an extent as in Example 1. This is not surprising since this system is of size $M = 2$ while the first was $M = 1$, as discussed in Section 5.2. The convergence rate of the error improves from close to $O(N^{-1/2})$ for the standard method, as expected, to $O(N^{-0.596})$ for the quasirandom method with 20 continuation steps.

As in Example 1, the plot against cpu time shows that the extra cpu time required for renumbering and continuation is more than made up for by the reduced errors. For the Helium atom, at an error level of about 10^{-3} we see a reduction in cpu time by a factor of almost 3 for quasirandom Diffusion Monte Carlo with 20 continuation steps.

It was found that taking more than 20 steps was not profitable in terms of error reduction versus extra cpu time for this example.

7 Conclusions

The results of the two examples above demonstrate that quasirandom Diffusion Monte Carlo is not only feasible, but can actually produce more accurate results in a given amount of cpu time than standard Diffusion Monte Carlo. It is the two techniques discussed above, renumbering and continuation, which make this possible.

There are many possible directions for further work in this area. One would be to better understand how and why renumbering works, and to prove conclusively that it does work for some well defined class of problems. Another would be to apply the methods described here to more complicated quantum mechanical systems involving perhaps 3 or 4 bodies in motion. A third direction would be to adapt the methods here to fermion problems in which the wavefunction changes sign, introducing new difficulties, as mentioned in Section 2.2. In addition, the problem of quasirandom error measurement needs to be further addressed, as mentioned in Section 6.1.

Finally, the results presented here should demonstrate that quasirandom sequences can perhaps be beneficial for a far wider range of Monte Carlo applications than presently thought possible.

References

[1] J. B. Anderson and C. A. Traynor. Quantum chemistry by random walk: Exact treatment of many-electron systems. Research Report, May 1991.

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

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

[4] D. Ceperly and B. Alder. Quantum Monte Carlo for molecules: Green's function and nodal release. *Journal of Chemical Physics*, 81(12):5833–5844, 1984.

[5] D. Ceperly and B. Alder. Quantum Monte Carlo. *Science*, 231:555–560, Feb. 1986.

[6] S. A. Chin. Quadratic diffusion Monte Carlo algorithms for solving atomic many-body problems. *Physical Review A*, 42(12):6991–7005, Dec. 1990.

[7] B. L. Fox. Implementation and relative efficiency of quasirandom sequence generators. *ACM Transactions of Mathematical Software*, 12(4):362–376, Dec. 1986.

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

[9] R. Grimm and R. Storer. Monte Carlo solution of Schrödinger's equation. *Journal of Compuational Physics*, 7:134–156, 1971.

[10] J. Halton. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, 2:84–90, 1960.

[11] S. Heinrich and A. 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.

[12] E. Helfand. Numerical integration of stochastic differential equations. *The Bell System Technical Journal*, pages 2289–2299, Dec. 1979.

[13] C. Joslin and S. Goldman. Quantum Monte Carlo studies of two-electron atoms constrained in spherical boxes. *Journal of Physics B - Atomic, Molecular, and Optical Physics*, 25(9):1965–1975, 1992.

[14] M. H. Kalos and P. A. Whitlock. *Monte Carlo Methods, Volume I: Basics.* J. Wiley and Sons, New York, 1986.

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

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

[17] M. E. Lee and K. E. Schmidt. Green's Function Monte Carlo. *Computers in Physics*, pages 192–197, March/April 1992.

[18] Y. L. Levitan, N. Markovich, S. Rozin, and I. Sobol′. Short communications on quasirandom sequences for numerical computations. *Zh. vŷchisl. Mat. mat. Fiz.*, 28(5):755–759, 1988.

[19] G. Marsaglia. Normal (Gaussian) random variables for supercomputers. *The Journal of Supercomputing*, 5:49–55, 1991.

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

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

[22] B. Moskowitz. Improved stochastic simulation using quasi-Monte Carlo: A computational study. *Mathematical and Computer Modelling*, submitted, Aug. 1994.

[23] B. Moskowitz and R. E. Caflisch. Smoothness and dimension reduction in quasi-Monte Carlo methods. *Mathematical and Computer Modelling*, submitted, Mar. 1994.

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

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

[26] W. H. Press and S. A. Teukolsky. Quasi- (that is, sub-) random numbers. *Computers in Physics*, pages 76–79, Nov/Dec 1988.

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

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

[29] I. Sobol′. Quasi-Monte Carlo methods. *Progress in Nuclear Energy*, 24:55–61, 1990.

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

[31] 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, Mar. 1991.

[32] S. Zhang and M. Kalos. Bilinear quantum Monte-Carlo - expectations and energy differences. *Journal of Statistical Physics*, 70(3-4):515–533, Feb. 1993.