A Fast and Efficient Algorithm for Slater Determinant Updates in Quantum Monte Carlo Simulations

Phani K.V.V. Nukala

Computer Science and Mathematics Division,

Oak Ridge National Laboratory, Oak Ridge, TN 37831-6164, USA

P. R. C. Kent

Center for Nanophase Materials Sciences,

Oak Ridge National Laboratory, Oak Ridge, TN 37831-6164, USA

Abstract

We present an efficient low-rank updating algorithm for updating the trial wavefunctions used in Quantum Monte Carlo (QMC) simulations. The algorithm is based on low-rank updating of the Slater determinants. In particular, the computational complexity of the algorithm is $\mathcal{O}(kN)$ during the k-th step compared with traditional algorithms that require $\mathcal{O}(N^2)$ computations, where N is the system size. For single determinant trial wavefunctions the new algorithm is faster than the traditional $\mathcal{O}(N^2)$ Sherman-Morrison algorithm for up to $\mathcal{O}(N)$ updates. For multideterminant configuration-interaction type trial wavefunctions of $M+1$ determinants, the new algorithm is significantly more efficient, saving both $\mathcal{O}(MN^2)$ work and $\mathcal{O}(MN^2)$ storage. The algorithm enables more accurate and significantly more efficient QMC calculations using configuration interaction type wavefunctions.

I. INTRODUCTION

Quantum Monte Carlo (QMC) is an approach capable of yielding highly accurate results in systems ranging from isolated molecules to the solid state[1]. The success of most common QMC methods, namely variational Monte Carlo (VMC) and diffusion Monte Carlo (DMC), depends crucially on the choice of trial wavefunction. Indeed, the trial wavefunction limits both the statistical efficiency and accuracy of the simulation. In QMC methods, the evaluation of the trial wavefunction becomes the most demanding part of the calculation especially when sufficiently large systems are considered or accurate simulations are required. This aspect of QMC was recognized even in the earliest DMC calculations, e.g. Ref. [2]. Consequently, the choice of trial wavefunction used in QMC calculations is motivated both by the accuracy and the speed of evaluation.

The most common form of trial wavefunction is of the Slater-Jastrow type

$$\Psi(\mathbf{R}) = D(\mathbf{R})e^{J(\mathbf{R})}, \tag{1}$$

where, neglecting spin, $D(\mathbf{R})$ is a Slater determinant, $J(\mathbf{R})$ is a Jastrow function, and $\mathbf{R} = \{\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_N\}$ is a vector of the position $\mathbf{r}_i$ of each electron. In QMC, the simulation commonly proceeds by proposing a local change to the electronic system configuration $\mathbf{R}$ to $\mathbf{R}'$. This local change in $\mathbf{R} \mapsto \mathbf{R}'$ is induced by the movement of one electron at a time from position $\mathbf{r}_i$ to $\mathbf{r}'_i$. The probability that the proposed local change is accepted is dependent on the transition probability, which depends on the ratio of $\Psi(\mathbf{R}')/\Psi(\mathbf{R})$, where $\mathbf{R}'$ is a new set of electron positions. This transition probability computation in turn requires the computation of the ratio of determinants $D(\mathbf{R}')$ and $D(\mathbf{R})$ in the new and old configurations respectively. Although a complete re-computation of $D(\mathbf{R}')$ can be made, an efficient algorithm that computes the necessary ratio $D(\mathbf{R}')/D(\mathbf{R})$ without resorting to a complete independent calculation of each determinant can significantly increase the overall efficiency of QMC simulations. Indeed, this efficiency measure is essential to the success of Slater-Jastrow wavefunctions; for a single electron move, the conventional algorithms use Sherman-Morrison formula (special case of Sherman-Morrison-Woodbury formula [3]) which reduces the cost of evaluating $D(\mathbf{R}')/D(\mathbf{R})$ to $\mathcal{O}(N)$, with an $\mathcal{O}(N^2)$ cost if the move is accepted, compared with $\mathcal{O}(N^3)$ for a naive evaluation of the determinant. Once the ratio of the determinants has been calculated, most quantities required in the Monte Carlo can be obtained through a simple multiplicative scaling[1]. Comparatively

recently, "linear scaling" approaches have been developed to reduce the cost of evaluating the determinants[4, 5, 6, 7, 8] by exploiting spatial locality in the studied physical system. In this paper, we explore alternative and complementary approaches to speedup the computation of transition probabilities and determinant ratios in QMC calculations.

The most accurate and commonly used QMC method is the DMC method performed in the fixed node approximation. This method exhibits a varational error in the energy depending on the quality of the nodal surfaces (zeroes) of the trail wavefunction. To improve the nodes as well as the variational quality of the trial function, it is now routine to utilize multiple determinant trial functions. These are commonly obtained from multiconfiguration quantum chemistry approaches such as the configuration interaction method where the ground state determinant D is supplemented by single and double excitations from the ground state. That is,

$$\Psi(\mathbf{R}) = \left[D(\mathbf{R}) + \sum_{a,c} \alpha_{a,c} D_a^c(\mathbf{R}) + \sum_{a,b,c,d} \beta_{a,b,c,d} D_{ab}^{cd}(\mathbf{R}) + ... \right] e^{J(\mathbf{R})}, \qquad (2)$$

where D_{ab}^{cd} denotes a double excitation with orbitals a and b replaced by c and d respectively, and α and β denote the multi-determinant expansion coefficients. Higher order excitations may be progressively included. Such an expansion of the wavefunction allows the nodal surface to be improved.

There are many strong motivations for minimizing the computational cost of multideterminant wavefunctions in QMC: Recent benchmark tests of the accuracy achievable in all electron VMC utilized, for example, up to 499 determinants to obtain over 90% of the correlation energy in the first row atoms[9]. To obtain a similar fraction of correlation energy in larger systems, more determinants are likely required. Numerous recent studies [10, 11, 12] have shown the utility of increased numbers of determinants for improved accuracy in atomic, molecular, and solid-state applications. In general this result is expected since quantum chemical techniques systematically improve the wavefunction with increased numbers of determinants. Improved trial wavefunctions using multideterminants are required for large systems such as the C_{60} fullerene where current trial wavefunctions are insufficient for computing accurate optical properties[13]. Multiple determinants may also be required to represent certain spin symmetries, e.g. Ref. [14]. Additionally, we have also recently shown that it is possible to sample the ground state wavefunction into a configu-

ration expansion[15] and subsequently improve the trial wavefunction[16]. This application requires the use of large configuration interaction expansions consisting of potentially thousands of determinants.

In this paper we propose an efficient algorithm for utilizing Slater-Jastrow trial wavefunctions in QMC simulations. The algorithm is particularly efficient for multideterminant wavefunctions. Extension to related alternative wavefunction forms such as multi-pfaffian and multi-backflow wavefunctions is straightforward. In Section II we present the details of the algorithm. Section III presents benchmark timing and efficiency measures for single determinant calculations using a variety of system sizes. The multideterminant case is analysed in Section IV. Conclusions are given in Section V.

II.   ALGORITHMS FOR UPDATING SLATER DETERMINANTS

As mentioned earlier, in QMC, the Monte Carlo simulation proceeds by proposing a local change to the electronic system configuration $\mathbf{R}$ to $\mathbf{R}'$. The acceptance criterion for each such local change follows the traditional Metropolis algorithm, which requires the computation of the transition probability. Each time a local change is accepted, the Slater matrix $\mathbf{D}(\mathbf{R})$ is updated to $\mathbf{D}(\mathbf{R}')$ by modifying one of the rows of $\mathbf{D}(\mathbf{R})$ corresponding to an electron movement from $\mathbf{r}$ to $\mathbf{r}'$. The simulation then proceeds by proposing a new local change, which requires the re-computation of the determinant of Slater matrix $\mathbf{D}(\mathbf{R}'')$ in the subsequent configuration $\mathbf{R}''$. This progression of the simulation via local changes typically proceeds for many thousands to millions of steps until observables such as the total energy converge to a desired statistical accuracy.

The slater matrix in configuration $\mathbf{R} = (\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_N)$ is given by

$$
\mathbf{D}(\mathbf{R}) = \begin{bmatrix} \phi_1(\mathbf{r}_1) & \phi_2(\mathbf{r}_1) & \cdots & \phi_N(\mathbf{r}_1) \\ \phi_1(\mathbf{r}_2) & \phi_2(\mathbf{r}_2) & \cdots & \phi_N(\mathbf{r}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(\mathbf{r}_N) & \phi_2(\mathbf{r}_N) & \cdots & \phi_N(\mathbf{r}_N) \end{bmatrix} \tag{3}
$$

where $\mathbf{r}_i$ and ϕ_i for $i = 1, 2, \ldots, N$ indicate respectively the spatial coordinates and spin-orbitals of i-th electron. Because we are moving a single electron (say p-th electron) at a time from position $\mathbf{r}_p \mapsto \mathbf{r}'_p$, the Slater matrix in the new electronic configuration $\mathbf{R}' =$

$(\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}'_p, \cdots, \mathbf{r}_N)$ is simply obtained by modifying the p-th row as

$$
\mathbf{D}(\mathbf{R}') = \begin{bmatrix}
\phi_1(\mathbf{r}_1) & \phi_2(\mathbf{r}_1) & \cdots & \phi_p(\mathbf{r}_1) & \cdots & \phi_N(\mathbf{r}_1) \\
\phi_1(\mathbf{r}_2) & \phi_2(\mathbf{r}_2) & \cdots & \phi_p(\mathbf{r}_2) & \cdots & \phi_N(\mathbf{r}_2) \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
\phi_1(\mathbf{r}'_p) & \phi_2(\mathbf{r}'_p) & \cdots & \phi_p(\mathbf{r}'_p) & \cdots & \phi_p(\mathbf{r}'_p) \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\phi_1(\mathbf{r}_N) & \phi_2(\mathbf{r}_N) & \cdots & \phi_p(\mathbf{r}_N) & \cdots & \phi_N(\mathbf{r}_N)
\end{bmatrix} \tag{4}
$$

The Metropolis probability to accept or reject the move is dependent on the ratio of determinants of Slater matrices $R = \frac{D(\mathbf{R}')}{D(\mathbf{R})}$, where $D(\mathbf{R}')$ is the determinant of Slater matrix $\mathbf{D}(\mathbf{R}')$ and $D(\mathbf{R})$ is the determinant of $\mathbf{D}(\mathbf{R})$. The transition probability is in general proportional to $|R|^2$, assuming real wavefunctions. If the move is accepted, then the system configuration changes to $\mathbf{R}'$; if not, the move is rejected and the system remains in configuration $\mathbf{R}$. In the following, whenever the context is clear, we denote $D(\mathbf{R}')$ by D' and $D(\mathbf{R})$ by D.

For the Monte Carlo simulation to be efficient, all quantities related to the transition probability and any observables must be computed with minimum computational operations. In the case of a single determinant wavefunction the ratio D'/D is required. For a single electron move this corresponds to a change of a single row in the Slater matrix. However, for the case of the multideterminant wavefunction, as in Eq. 2, all ratios $D_a^{c'}/D_a^c$ and $D_{ab}^{cd'}/D_{ab}^{cd}$ are required. These ratios involve determinants with both orbital replacements and single electron moves (i.e., both row and column changes) when compared to the original ground state determinant D.

For a single electron move, the basic computational problem involved during the $(k+1)$-th MC step may be expressed as: *Given the determinanat D_k of Slater matrix $\mathbf{D}_k$, compute the determinant D_{k+1} of $\mathbf{D}_{k+1}$ such that*

$$
\mathbf{D}_{k+1} = \mathbf{D}_k + \mathbf{e}_{\mathbf{p}(k)} \mathbf{v}_k^t \tag{5}
$$

where $\mathbf{p}(k)$ defines an index vector that maps $k \mapsto p$ such that $\mathbf{p}(k) = p$, and $\mathbf{e}_p$ denotes an unit vector with 1 on the p-th entry and 0 everywhere else. The vector $\mathbf{v}_k$ corresponds to the *change* in Slater matrix due to the displacement of the p-th electron during the $(k+1)$-th MC step and is given by

$$
\mathbf{v}_k = \left(\phi_1(\mathbf{r}'_p) - \phi_1(\mathbf{r}_p), \cdots, (\phi_N(\mathbf{r}'_p) - \phi_N(\mathbf{r}_p)) \right)^t \tag{6}
$$

A straight-forward computation of D_{k+1} may be obtained as

$$D_{k+1} = (1 + \mathbf{v}_k^t \mathbf{D}_k^{-1} \mathbf{e}_{\mathbf{p}(k)}) \, D_k \tag{7}$$

and $R_k = \frac{D_{k+1}}{D_k}$ can be evaluated as

$$R_k = (1 + \mathbf{v}_k^t \mathbf{D}_k^{-1} \mathbf{e}_{\mathbf{p}(k)}) \tag{8}$$

Hence, for any given k, R_k can be evaluated efficiently in $\mathcal{O}(N)$ computations since $\mathbf{D}_k^{-1} \mathbf{e}_{\mathbf{p}(k)}$ can be interpreted as the p-th column of $\mathbf{D}_k^{-1}$, i.e., $\mathbf{D}_k^{-1}(:, p) = \mathbf{D}_k^{-1} \mathbf{e}_{\mathbf{p}(k)}$. However, repetitive computation of R_k during each of the MC simulation steps (for $k = 0, 1, 2, \ldots$) requires an efficient procedure to compute $\mathbf{D}_k^{-1}$ for each k. For this purpose, traditional algorithms employ the Sherman-Morrison formula to update $\mathbf{D}_k^{-1} \mapsto \mathbf{D}_{k+1}^{-1}$, which can be expressed as

$$\mathbf{D}_{k+1}^{-1} = \mathbf{D}_k^{-1} - \frac{\mathbf{D}_k^{-1} \mathbf{e}_{\mathbf{p}(k)} \mathbf{v}_k^t \mathbf{D}_k^{-1}}{(1 + \mathbf{v}_k^t \mathbf{D}_k^{-1} \mathbf{e}_{\mathbf{p}(k)})} \tag{9}$$

Using this formula, $\mathbf{D}_k^{-1}$ can be updated to $\mathbf{D}_{k+1}^{-1}$ in $\mathcal{O}(N^2)$ computations. However, since the required number of MC steps in a typical Monte Carlo simulation readily extends to the thousands to millions range, and can increase with increasing system sizes, $\mathcal{O}(N^2)$ scaling of these traditional algorithms poses a significant hindrance for the simulation of large system sizes despite the fact that such large scale simulations are necessary to develop a better understanding of relevant chemistry and physics. As discussed in the introduction, Sec I, multiple determinants compound this problem.

Alternatively, an efficient recursive algorithm for computing $\mathbf{D}_{k+1}^{-1}$ may be formulated by expressing Eq. 9 as

$$\begin{aligned} \mathbf{D}_{k+1}^{-1} &= \left[\mathbf{I} - \frac{\mathbf{D}_k^{-1} \mathbf{e}_{\mathbf{p}(k)} \mathbf{v}_k^t}{(1 + \mathbf{v}_k^t \mathbf{D}_k^{-1} \mathbf{e}_{\mathbf{p}(k)})}\right] \mathbf{D}_k^{-1} \\ &= \left[\mathbf{I} - \gamma_k \mathbf{u}_k \mathbf{v}_k^t\right] \mathbf{D}_k^{-1} \end{aligned} \tag{10}$$

where $\mathbf{u}_k = \mathbf{D}_k^{-1} \mathbf{e}_{\mathbf{p}(k)}$ and $\gamma_k = \frac{1}{R_k}$. Based on Eq. 10, a recursive scheme for computing $\mathbf{D}_{k+1}^{-1}$ may be formulated as

$$\begin{aligned} \mathbf{D}_{k+1}^{-1} &= (\mathbf{I} - \gamma_k \mathbf{u}_k \mathbf{v}_k^t) \ldots (\mathbf{I} - \gamma_0 \mathbf{u}_0 \mathbf{v}_0^t) \, \mathbf{D}_0^{-1} \\ &= \left[\prod_{j=0}^{k} (\mathbf{I} - \gamma_j \mathbf{u}_j \mathbf{v}_j^t)\right] \mathbf{D}_0^{-1} \end{aligned} \tag{11}$$

An $\mathcal{O}(kN)$ recursive algorithm based on Eq. 11 is presented in Algorithm 1. For each additional step k, this algorithm requires storage space for two vectors $\mathbf{u}_k$ and $\mathbf{v}_k$ of size N. In addition, we need to store an index vector $\mathbf{p}(k)$ that maps $k \mapsto p$ such that $\mathbf{p}(k) = p$.

III.   SINGLE DETERMINANT BENCHMARKS

In order to compare the computational efficiency of the recursive algorithm with the traditional algorithm, we first tested the case of a single determinant wavefunction. An analysis of the multideterminant case is given in Sec. IV.

We tested the algorithms on a randomly generated matrix $\mathbf{D}_0$. That is, since the algorithms are applicable for general matrices, we start with a matrix $\mathbf{D}_0$ whose elements are randomly chosen between zero and one. Then we consider rank-1 updates of $\mathbf{D}_0$ as given by Eq. 5 for m number of steps. The site locations p are chosen sequentially, modulo N, for these m steps. The updated orbitals are chosen randomly. Figure 1 presents the ratio of the computational timings obtained using the full matrix updating and recursive updating algorithms. The timings were obtained using a standalone benchmark code using double precision arithmetic. We used the same data structures both in our recursive and full QMC simulations. Machine optimized linear algebra library calls were used for both algorithms. Timings were obtained on a 2.73 GHz Intel Xeon processor with 12 MB Cache.

Examining the timings shown in Fig. 1, we see that the recursive update algorithm is always significantly faster than the Sherman-Morrison algorithm for a small number of updates. For up to ten updates, the new algorithm ~ 10 times faster for a 100 sized matrix, while for a 6400 sized matrix the new algorithm is ~ 1000 times faster. For increased numbers of updates the ratio of timings decreases. The crossover between the two algorithms occurs near the theoretically expected $k = N$ updates.

Due to the iterative nature of both algorithms numerical errors accumulate over time. It is common practise in QMC simulations to fully recalculate the inverse cofactor matrices from time to time to limit these errors. Such a recalculation requires $\mathcal{O}(N^3)$ operations. We have compared the numerical errors of the recursive update algorithm with the Sherman-Morrison algorithm and find the performance to be similar. Figure 2 illustrates the build up of errors for both algorithms for a single run.

Figure 2 shows that both algorithms have good stability and on average give high accu-

racy, particularly for small numbers of updates. However, for both algorithms the average and maximum numerical error in the determinant ratio gradually increases with the number of updates and can become substantial. In both cases the maximum error for a fixed number of updates can deviate by several orders of magnitude from the average. This behavior appears to be due to the occasional mixture of very small and very large numbers in the update formulae which results in a significant loss of precision. This data shows that while the recursive algorithm performs similarly to the Sherman-Morrison algorithm, it is vital to check sufficient accuracy is obtained if large numbers of updates are performed.

IV. MULTIPLE DETERMINANT WAVEFUNCTIONS

In the case of multiple determinant wavefunctions such as a configuration interaction expansion, all the excited Slater matrices $\mathbf{D}_a^c(\mathbf{R})$ and $\mathbf{D}_{ab}^{cd}(\mathbf{R})$ are similar to the ground state matrix $\mathbf{D}(\mathbf{R})$, and differ only by a few column interchanges. The use of the recursive algorithm provides an efficient way of calculating the transition probability compared to the traditional algorithm; It is not only faster but also requires reduced storage of $\mathcal{O}(N^2)$ for storing *only* $\mathbf{D}_0^{-1}$ *of the ground state matrix*. No other potentially large data must be stored, although it is advantageous to reuse the current determinant values between MC steps. The recursive algorithm is used to compute the non-ground state determinants via column changes to the ground state matrix. The cost of each particle move is constant and does not increase when many steps are taken.

For simplicity we analyse the case of a multiple determinant wavefunction consisting of only the ground state determinant and M determinants doubly excited from this state. Conventionally the $\mathbf{D}_0^{-1}$ as well as all the excited Slater matrix inverses are stored in memory to enable fastest possible update using the traditional algorithm. When the recursive algorithm is applied to multiple determinant wavefunctions, we store only the $\mathbf{D}_0^{-1}$ of the ground state. Conventional updates are performed on this determinant and the recursive algorithm is used to compute the other excited determinants since the excited and ground state Slater matrices differ by a few column changes. Note that successive row updates can be performed in $\mathcal{O}(kN)$ operations using an algorithm similar to that of Algorithm 1. However, successive row updates followed by multiple column updates always requires a $\mathcal{O}(N^2)$ cost associated with a matrix-vector multiplication. Since proposed moves are usu-

8

ally accepted in DMC calculations with an acceptance ratio of $> 99\%$, it is convenient to use the conventional (Sherman-Morrison) algorithm to update the inverse of the ground state Slater matrix. It should also be noted that in the event the proposed move is accepted, the traditionally updated $D(\mathbf{R}')$ used in the evaluation of the excited state determinants can be reused: the recursive update algorithm then requires no additional $\mathcal{O}(N^2)$ work over a single determinant calculation. Consequently, using the recursive algorithm an M determinant wavefunction can be used with an updating cost scaling only linearly in M and system size N compared to an N^2 scaling cost using the traditional algorithm.

To evaluate determinant ratios such as $D_{ab}^{cd}(\mathbf{R}')/D_{ab}^{cd}(\mathbf{R})$ we first perform a traditional update to obtain $D(\mathbf{R}')$. The recursive algorithm is then used to compute $D_{ab}^{cd}(\mathbf{R}')$ from $D(\mathbf{R}')$. We assume that $D_{ab}^{cd}(\mathbf{R})$ is stored and available from a previous MC step, but this can also be calculated using two applications of the recursive algorithm to $D(\mathbf{R})$. In Table I we compare the costs of evaluating the determinant ratios in $\Psi(\mathbf{R}')/\Psi(\mathbf{R})$. Independent of the amount of storage chosen for the traditional scheme, the recursive scheme displays an improved computational cost by a factor $\mathcal{O}(MN^2)$, or M times the cost of a complete single determinant update. The single determinant benchmarks of Sec. III show that these updates, which are few in number and hence correspond to the left side of Fig. 1, are several orders of magnitude faster than the traditional algorithm.

V.   CONCLUSIONS

In this paper, we presented an efficient low-rank updating algorithm for QMC simulations. The algorithm requires only $\mathcal{O}(kN)$ computations during k-th MC step compared with $\mathcal{O}(N^2)$ computations required by traditional algorithms. Our numerical simulations indicate that for small numbers of updates of a single determinant this algorithm is orders of magnitude faster than traditional algorithms. For single determinant wavefunctions, the traditional algorithms remain the preferred choice for more than $\mathcal{O}(N)$ updates. For multideterminant wavefunctions of $M + 1$ determinants, our algorithm is the preferred choice, being significantly faster and of particular interest for large systems. In addition, it enables workspace to be reduced by a factor $\mathcal{O}(MN^2)$. The speed and storage savings of this new algorithm enables QMC calculations to use thousands of determinants.

Acknowledgment

[1] W. M. C. Foulkes, L. Mitas, R. J. Needs and G. Rajagopal, Rev. Mod. Phys. **73**, 33 (2001).

[2] P. J. Reynolds, D. M. Ceperley, B. J. Alder, and W.A. Lester, J. Chem. Phys. **77**, 5593 (1982).

[3] G. H. Golub and C. F. V. Loan, *Matrix Computations* (The Johns Hopkins University Press, Baltimore, 1996).

[4] A. J. Williamson, R. Q. Hood, and J. C. Grossman, Phys. Rev. Lett. **87**, 246406 (2001).

[5] F. A. Reboredo and A. J. Williamson, Phys. Rev. B **71**, 121105 (2005).

[6] D. Alfe and M. J. Gillan, Journal of Physics: Cond. Mat. **16**, L305 (2004).

[7] A. Aspuru-Guzik, R. Salomon-Ferrer, B. Austin, and W. A. Lester Jr, J. Comp. Chem. **26**, 708-715 (2005).

[8] J. Kussmann, H. Riede, and C. Ochsenfeld, Phys. Rev. B **75**, 165107 (2007).

[9] M. D. Brown, J. R. Trail, P. L. Rios, and R. J. Needs, J. Chem. Phys. **126**, 224110 (2007).

[10] J. Lawson, C. Bauschlicher, J. Toulouse, C. Filippi, and C. Umrigar, Chem. Phys. Lett. **466**, 170 (2008).

[11] J. Toulouse and C. J. Umrigar, J. Chem. Phys. **128**, 174101 (2008).

[12] J. R. Trail and R. J. Needs, J. Chem. Phys. **128**, 204103 (2008).

[13] M. L. Tiago, P. R. C. Kent, R. Q. Hood, and F. A. Reboredo, J. Chem. Phys. **129**, 084311 (2008).

[14] R. Q. Hood, P. R. C. Kent, R. J. Needs, and P. R. Briddon, Phys. Rev. Lett. **91**, 076403 (2003).

[15] F. A. Reboredo and P. R. C. Kent, Phys. Rev. B **77**, 245110 (2008).

[16] F. A. Reboredo, R. Q. Hood, and P. R. C. Kent, Accepted in Physical Review B (2009), manuscript BV10869.

Algorithm 1 Recursive Algorithm (p-th electron moves)

1: Given $\mathbf{D}_0^{-1}$ and $\mathbf{v}_k$

2: Set $\mathbf{p}(k) = p$

3: Set $\mathbf{u}_k = \mathbf{D}_0^{-1}\mathbf{e}_{\mathbf{p}(k)} = \mathbf{D}_0^{-1}(:,p)$

4: **for** $i = 0$ to $k - 1$ **do**

5: Compute $\mathbf{u}_k = \mathbf{u}_k - \gamma_i \ (\mathbf{v}_i^t\mathbf{u}_k) \ \mathbf{u}_i$

6: **end for**

7: Compute $R_k = 1 + \mathbf{v}_k^t\mathbf{u}_k$

8: **if** Accept **then**

9: Compute $\gamma_k = \frac{1}{R_k}$

10: Save $\mathbf{u}_k$, $\mathbf{v}_k$ and γ_k
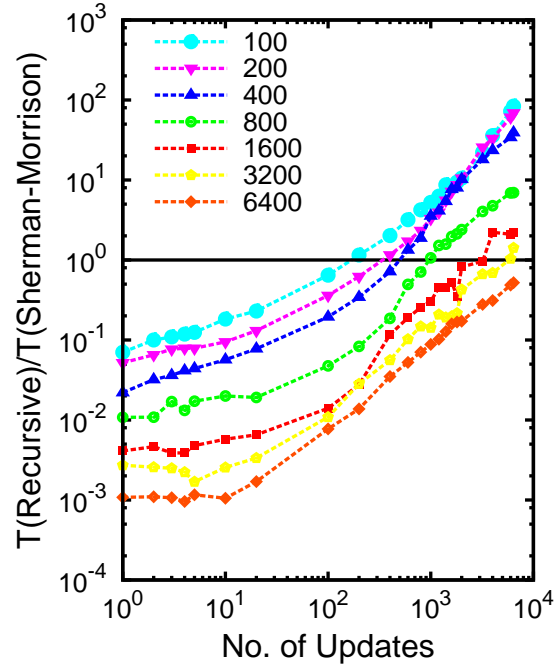
11: k = k+1

12: **end if**

FIG. 1: Relative timing of the recursive update algorithm to the traditional Sherman-Morrison algorithm for different matrix sizes. Ratios less than one indicate that the recursive algorithm is faster.
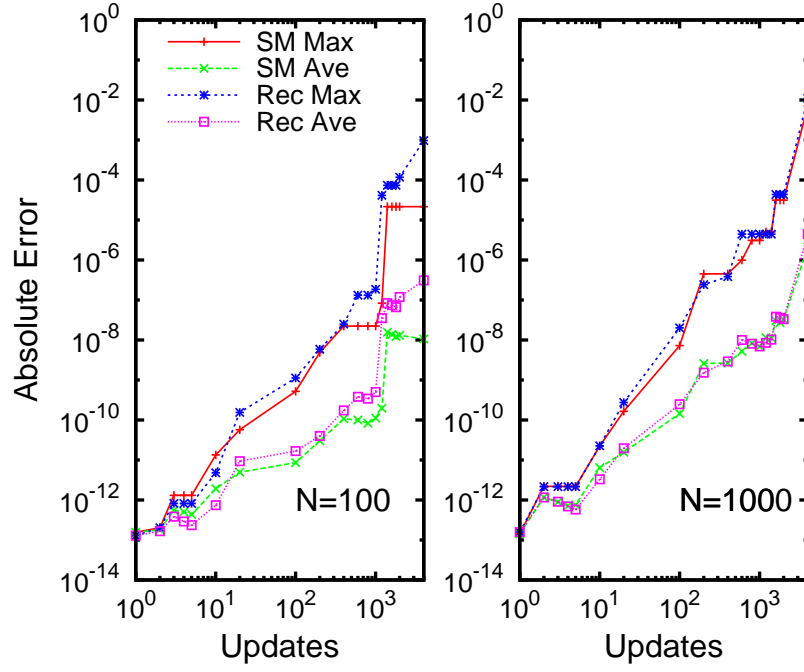
FIG. 2: Absolute numerical errors in computed determinant ratios using the recursive update (Rec) and Sherman-Morrison (SM) algorithms with double precision arithmetic for matrix sizes of 100 (left) and 1000 (right). The behavior of the algorithms is similar.

14

Algorithm	Move evaluation cost	Move acceptance cost	Storage cost
Traditional	$\mathcal{O}((1+M)N)$	$\mathcal{O}((1+M)N^2)$	$\mathcal{O}(2(1+M)N^2)$
Minimum storage traditional	$\mathcal{O}((1+2M)N^2+(1+M)N)$	$\mathcal{O}(N^2)$	$\mathcal{O}(2N^2)$
Recursive	$\mathcal{O}(N^2+3MN)$	$\mathcal{O}(N^2)$	$\mathcal{O}(2N^2)$

TABLE I: Cost of computing wavefunction ratios using traditional and recursive algorithms for proposed and accepted single electron moves. The wavefunction is of the configuration interaction doubles type consisting of M double excitations from a single ground state determinant. For the storage costs we consider only the most significant $\mathcal{O}(N^2)$ and higher contributions. For at least the ground state determinant, both the full matrix and its inverse are stored resulting in the lead factor of 2 in the storage costs. In the traditional algorithm, the emphasis is on speed and hence all the excited state matrices and ground state matrix (along with its inverses) are stored. For the "minimum storage traditional", the emphasis is on limiting storage costs even at the expense of increased computational cost. Hence, in the "minimum storage traditional" algorithm, we assume that the traditional algorithm is used but only the ground state matrix is stored and the remaining matrices are computed based on the ground state matrix.