PHYSICAL REVIEW B **81**, 081103(R) (2010)

Efficient matrix-product state method for periodic boundary conditions

Peter Pippan,[1] Steven R. White,[2] and Hans Gerd Evertz[1]
[1]*Institut für Theoretische Physik, Technische Universität Graz, A-8010 Graz, Austria*
[2]*Department of Physics and Astronomy, University of California, Irvine, California 92697, USA*
(Received 30 December 2009; published 12 February 2010)

We introduce an efficient method to calculate the ground state of one-dimensional lattice models with periodic boundary conditions. The method works in the representation of matrix product states (MPS), related to the density matrix renormalization group (DMRG) method. It improves on a previous approach by Verstraete *et al.* We introduce a factorization procedure for long products of MPS matrices, which reduces the computational effort from m^5 to m^3, where m is the matrix dimension, and $m \simeq 100-1000$ in typical cases. We test the method on the $S=\frac{1}{2}$ and $S=1$ Heisenberg chains. It is also applicable to nontranslationally invariant cases. The method makes ground-state calculations with periodic boundary conditions about as efficient as traditional DMRG calculations for systems with open boundaries.

DOI: 10.1103/PhysRevB.81.081103 PACS number(s): 71.27.+a, 05.10.Cc, 02.70.−c, 75.10.Pq

One of the most severe problems in condensed-matter theory is the exponential growth of the Hilbert space with system size. This limits many methods such as exact diagonalization. One strategy that overcomes these difficulties is to approximate the ground state in some reduced Hilbert space.

The *density matrix renormalization group* (DMRG) (Refs. 1–3) is one prominent example of such methods. By tracing out "unimportant" degrees of freedom, the real ground state is approximated in a much smaller space. DMRG works much better for open boundary conditions (obcs) than for periodic boundary conditions (pbcs). In the worst case where the correlation length is much smaller than the system size, if the obc system needs m_{obc} states per block for a given accuracy, the pbc system needs $O(m_{\mathrm{obc}}^2)$. Since the calculation time scales as m^3, the comparable time for pbc is $O(m_{\mathrm{obc}}^6)$. However, systems with obc naturally suffer from edge effects such as Friedel oscillations. An efficient method for pbc would be highly desirable. For example, it would make finite-size scaling easier, and allow the direct representation of finite momentum eigenstates.[4–6]

It can be shown that the ground state produced by DMRG can quite naturally be written in terms of a so-called *matrix-product state* (MPS) (Refs. 4 and 5) for both obc and pbc. The original work presented an inefficient method for computing the MPS, which could not compete with DMRG. Recently, a number of new algorithms utilizing the MPS state directly have been introduced which are efficient and greatly extend the reach of DMRG/MPS techniques,[6–13] including the simulation of random systems or a generalization to two-dimensional (2D) systems. In the present Rapid Communication we investigate an algorithm presented in Ref. 8 for an MPS treatment of pbc systems. Within this approach $m_{\mathrm{pbc}} \approx m_{\mathrm{obc}}$, a tremendous improvement. However, that algorithm has a computational cost of m^5, making the net improvement modest.

Here we introduce an improvement to this pbc MPS algorithm based on the approximation of long products of certain large $(m^2 \times m^2)$ transfer matrices in terms of a singular value decomposition (SVD) with only a few singular values. A new circular update procedure allows us to work exclusively with such long products. Our approach improves the scaling of the algorithm dramatically to m^3.

MPS with pbc. We summarize the algorithm presented in Ref. 8 and explain some practical aspects. The ground state of a quantum-mechanical system such as a spin model, defined on a one-dimensional lattice of N sites, can be written in terms of an MPS (Ref. 14)

$$|\phi\rangle = \sum_{s_1,s_2\cdots s_N} \mathrm{Tr}(A_{s_1}^{[1]} A_{s_2}^{[2]} \cdots A_{s_N}^{[N]}) |s_1 s_2 \cdots s_N\rangle, \qquad (1)$$

where $A_{s_i}^{[i]}$ are sets of d matrices of dimension $m \times m$ and d is the dimension of the Hilbert space of a single spin s_i. The trace in Eq. (1) ensures periodic boundary conditions. Any state can be written in this form if m is large enough; the power of the approach comes from the property that modest m produces excellent approximations to ground states of local Hamiltonians. Of course the expression above is purely formal and we need a procedure to optimize the matrices $A_{s_i}^{[i]}$. For any operator O_i on a site i we define the $m^2 \times m^2$ matrix[4]

$$E_{O_i}^{[i]} = \sum_{s,s'} \langle s|O_i|s'\rangle A_{s_i}^{[i]} \otimes (A_{s_i'}^{[i]})^*. \qquad (2)$$

Using these generalized transfer matrices, expectation values of products of operators can be easily evaluated as

$$\langle\phi|O_1 O_2 \cdots O_N|\phi\rangle = \mathrm{Tr}(E_{O_1}^{[1]} E_{O_2}^{[2]} \cdots E_{O_N}^{[N]}). \qquad (3)$$

The Hamiltonian can also be written using the relation above and the matrices $A_{s_i}^{[i]}$ can be optimized one by one in order to minimize the energy. Consider the Ising model $H=\Sigma_i \sigma_i^z \otimes \sigma_{i+1}^z$. To optimize matrices $A_{s_i}^{[i]}$ at site i, an effective Hamiltonian containing only matrices $A^{[1]} \cdots A^{[i-1]}$ and $A^{[i+1]} \cdots A^{[N]}$ can be constructed as follows:

$$H_{eff} = \mathbb{1}^s \otimes \tilde{h}^i + \sigma^z \otimes {}^{i+1}\tilde{\Sigma}_l^{i-1} + \sigma^z \otimes {}^{i+1}\tilde{\Sigma}_r^{i-1}, \qquad (4)$$

where $\mathbb{1}^s$ is the identity matrix in spin space and

$$h^i = \sum_k E_{\mathbb{1}}^{[i+1]} \cdots E_{\mathbb{1}}^{[k-1]} E_{\sigma^z}^{[k]} E_{\sigma^z}^{[k+1]} E_{\mathbb{1}}^{[k+2]} \cdots E_{\mathbb{1}}^{[i-1]},$$

$$ {}^{i+1}\Sigma_l^{i-1} = E_{\sigma^z}^{[i+1]} E_{\mathbb{1}}^{[i+2]} E_{\mathbb{1}}^{[i+3]} \cdots E_{\mathbb{1}}^{[i-1]}, $$

$$ {}^{i+1}\Sigma_r^{i-1} = E_{\mathbb{1}}^{[i+1]} E_{\mathbb{1}}^{[i+2]} \cdots E_{\mathbb{1}}^{[i-2]} E_{\sigma^z}^{[i-1]}. \quad (5) $$

In the equation above, all indices are taken modulo N. The tilde in Eq. (4) refers to the exchange of indices $X_{(ij)(kl)} = \tilde{X}_{(ik)(jl)}$. Together with a map of the identity matrix $N_{eff} = \mathbb{1}^s \otimes \tilde{N}^i$, $N^i = E_{\mathbb{1}}^{[i+1]} \cdots E_{\mathbb{1}}^{[N]} E_{\mathbb{1}}^{[1]} \cdots E_{\mathbb{1}}^{[i-1]}$, a new set of d matrices $A_{s_i}^{[i]}$ for fixed i is found by solving the generalized eigenvalue problem

$$ H_{eff}\,\mathrm{Vec}(A) = \epsilon N_{eff}\,\mathrm{Vec}(A), \quad (6) $$

with ϵ the expectation value of the energy and $\mathrm{Vec}(A)$ the dm^2 elements of $A_{s_i}^{[i]}$, aligned to a vector.

When a new set of matrices has been found, the matrices need to be regauged, in order to keep the algorithm stable. In DMRG this is not necessary since the basis of each block is orthogonal. The orthogonality constraint reads as $\Sigma_{s_i} A_{s_i}^{[i]} (A_{s_i}^{[i]})^\dagger = \mathbb{1}$. It can be satisfied as follows: The state is left unchanged when we substitute $A_s^{[l]} \to A_s^{[l]} X \equiv U^{[l],s}$ and $A_s^{[l+1]} \to X^{-1} A_s^{[l+1]}$, with some nonsingular matrix X. This matrix X has to be found such that $U_s^{[l]}$ obeys the normalization condition $\Sigma_s U_s^{[l]} (U_s^{[l]})^\dagger = \mathbb{1}$. We obtain X by calculating the inverse of the square root of $Q = \Sigma_s A_s^{[l]} (A_s^{[l]})^\dagger$. Since Q is not guaranteed to be nonsingular, the pseudoinverse has to be used,[15] by discarding singular values close to zero in an SVD of Q. H_{eff}^i can be calculated iteratively,[8] while updating the A-matrices one site at a time. One sweeps back and forth in a DMRG-like manner.

Vidal introduced a different approach, for infinitely long translationally invariant systems.[11] By assuming only two different kinds of matrices $A^{[1]}$ and $A^{[2]}$ and aligning them in alternating order, an algorithm for both ground state and time evolution can be constructed that updates the matrices in only $O(m^3)$ steps. However, unlike the periodic MPS method discussed here, Vidal's method does not apply to nontranslationally invariant systems (e.g., when impurities or a site-dependent magnetic field are studied). In addition, the periodic MPS method can be adapted[6] to treat excited states, whereas the method of Ref. 11 probably cannot, since the excitations would be spread over an infinite lattice and would have no effect on any individual site. Recently, a related approach came to our attention,[16] in which the E matrix of a translationally invariant system is treated in $O(m^3)$. Also recently, related Quantum Monte Carlo variational methods using tensor product states were introduced,[12,13] with scaling $O(Nm^3)$ per Monte Carlo sweep.

Computational efficiency. It was shown in Ref. 8 that the m needed for pbc systems in the MPS approach is comparable to the m needed in obc systems within DMRG. However, it is also vital how CPU time scales with m. In efficient DMRG programs, most operations can be done by computing multiplications of $m \times m$ matrices (see Ref. 3, Ch. II.i).

In contrast, in the MPS algorithm described above, operations on $m^2 \times m^2$ matrices need to be done to form the products of E-matrices that represent the Hamiltonian. So one would expect the algorithm to be of order $O(m^6)$. By taking advantage of the special form of the E matrices Eq. (2), multiplications can be done in $O(m^5)$ which is, however, still $O(m^2)$ slower than DMRG.

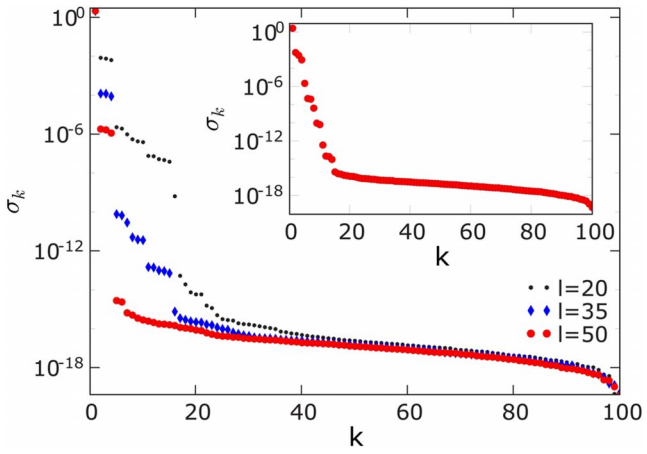

FIG. 1. (Color online) SVD of a product $E_{\mathbb{1}}^{[1]} \cdots E_{\mathbb{1}}^{[l]}$ with $m = 10$. The logarithm of the singular values σ_k is shown for different l in the case of a spin 1 Heisenberg chain of length 100 with periodic boundary conditions. The inset shows data for a spin $\frac{1}{2}$ Heisenberg chain.

Decomposition of products. We now introduce an approximation in the space of $m^2 \times m^2$ matrices which reduces the CPU time dramatically while the accuracy of the calculation does not suffer. Let us perform a singular value decomposition of a long product of E matrices

$$ E_{O_1}^{[1]} E_{O_2}^{[2]} \cdots E_{O_l}^{[l]} = \sum_{k=1}^{m^2} \sigma_k \mathbf{u}_k \mathbf{v}_k^T. \quad (7) $$

It turns out that the singular values σ_k decay very fast. This is shown in Fig. 1 for products of the form $\Pi_{i=1}^{l} E_1$ with various values of l, for the case of the spin 1 Heisenberg chain. One can see that the longer the product the faster the singular values decay, roughly exponentially in the length l. We therefore propose to approximate long products in a reduced basis

$$ \prod_{i=1}^{l} E_{O_i}^{i} \approx \sum_{k=1}^{p} \sigma_k \mathbf{u}_k \mathbf{v}_k^T, \quad (8) $$

with p chosen suitably large. In the example of Fig. 1, we would choose p to be 4 at $l = 50$. Remarkably, for longer products p can be as small as 2 without a detectable loss of accuracy. Thus, the large distance behavior of the ground state of the spin 1 chain is encoded in these two numbers, similar to the transfer matrices of a classical spin chain. The situation does not change significantly when more complicated operators such as the Hamiltonian are decomposed. Of course, the decay of the singular values will be model dependent. For a spin $\frac{1}{2}$ Heisenberg chain we found that the decomposition can be done in the same manner with approximately the same number of singular values to be kept.

A multiplication of a product with a new E matrix can therefore be done[17] in $O(pm^3)$ and a multiplication of two terms such as Eq. (8) can be done in $O(pp'm^2)$. By building

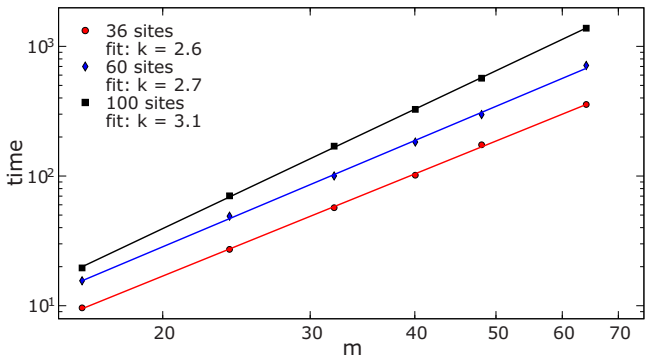

FIG. 2. (Color online) Scaling of the circular version of the algorithm. CPU time per sweep (one update of each site) is measured on a 100 site spin $\frac{1}{2}$ Heisenberg chain for different system sizes. The time is fitted to a function m^k.

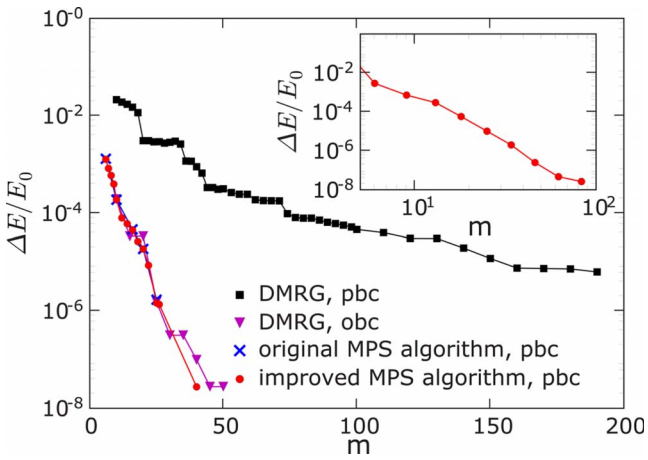

FIG. 3. (Color online) Relative error of the ground-state energy of the spin 1 Heisenberg model versus the dimension m of the reduced Hilbert space. DMRG results with obc and pbc are shown, as well as Matrix Product State results with pbc. The inset shows MPS results with pbc for a spin $\frac{1}{2}$ Heisenberg chain of 100 sites.

the effective Hamiltonian out of products in this representation, the iterative evaluation of the eigenvalue problem can be accelerated. Whereas in a dense form each matrix-vector multiplication,—which occurs in eigenvalue routines such as Lanczos or Davidson—takes $(dm^2)^2$ operations, it can now be done in $O(d^2pm^2)$. Note that all operations are now done on matrices of size $m \times m$.

Performing the SVD in m^3. A crucial step is the efficient generation of the SVD representation of a large $m^2 \times m^2$ matrix M in only $O(m^3)$ operations. We describe a simple algorithm, with a fixed number of singular values (four) to keep the notation simple. Suppose that $M=UdV$, with d a 4×4 diagonal matrix, and that multiplication of M by a vector (without using the SVD factorization) can be done in $O(m^3)$. To construct U, d, and V with $O(m^3)$ operations, we first form a random $4\times m^2$ matrix x, and construct $y=xM$. The 4 rows of y, are linear combinations of the rows of V. Orthonormalize them to form y'. Its rows act as a complete orthonormal basis for the rows of V. This means that $V=Vy'^Ty'$, and thus $M=My'^Ty'$. Construct $z=My'^T$, and perform an SVD on z: $z=UdV'$. Then $M=zy'=UdV$, where $V=V'y'$. V is row orthogonal because V' is orthogonal and y' is row orthogonal. The calculation time for the orthogonalization of y and the SVD of z is $O(m^2)$, and so the calculation time is dominated by the two multiplications by M, e.g., roughly $2\times4\times O(m^3)$.[18]

In applying this approach to the periodic MPS algorithm, M is a product of $O(N)$ E matrices such as in Eq. (5), which in turn are outer products (2). The multiplication with M can be done iteratively in $O(Nm^3)$ operations, analogously to the construction of H^i_{eff}. The calculation time is thus $O(Nm^3)$ for each SVD representation generated this way. It is only needed a few times per sweep (see below).

A circular algorithm. A speedup in the simulation can only be expected if the number of singular values that need to be included is sufficiently small. However, in the algorithm of Ref. 8 one sweeps back and forth through the lattice, so that close to the turning points, products of only a few E matrices appear, which require more singular values (Fig. 1). In the extreme case of only one E matrix, we would have $p=m^2$. To overcome this bottleneck we propose a modified method which proceeds through the chain in a *circular* fashion, thus making natural use of the periodic boundary conditions. Note that we cannot employ multiplications with inverse matrices E_O^{-1}, since they are too expensive to calculate. We consider the lattice as a circular ring, and divide it into thirds, or "sections." We perform update steps for one section at a time. To start one section, we first construct the Hamiltonian and other necessary operators [see Eq. (5)] corresponding to the other two sections of the lattice. Only a few such operators are needed. Each of them contains products of $N/3$ E matrices and is computed by an SVD decomposition as described before.

Then a set of these operators is made by successively adding sites from the right most part of the current section to the operators constructed for the section on the right, working one's way to the left. Adding a site involves the multiplication of an E-matrix to the left of an operator. These steps can each be done in $O(m^3)$ operations. When one has reached the left side of the current section, its initialization is finished and one can start the normal update steps, now building up a set of operators from the left, again in $O(m^3)$ operations. One stops when one reaches the right-hand side of the current section. Then the procedure repeats with the section to the right as the new current section. Some of the operators previously computed can be reused. The updates now go in a circular pattern rather than the usual back and forth.

By proceeding in this way on a system of length N, the blocks on which we have to perform an SVD are of length at least $N/3$ (if we split our system into three parts), so that the SVD will have only few singular values. Consequently, the algorithm is expected to scale like $O(Nm^3)$.

Test and results. To test our improvements, we studied spin 1 and spin $\frac{1}{2}$ Heisenberg chains up to length $N=100$. The exact ground-state energy for pbc on an $N=100$ chain in the spin 1 case is found to be $E_0/N \cong -1.401\,484\,038\,6(5)$ via a DMRG calculation with $m=2000$. The error is generously estimated from the truncation error and an extrapolation in m. The periodic result differs from the infinite system

result (determined using long open chains) only in the last decimal place, so we will call this value "exact."

We discarded singular values smaller than a 10^{-11}th of the largest one. This parameter is chosen such that the algorithm remains stable, which is not the case if the error bound is chosen too large (10^{-8} or larger). To decrease the time it takes until convergence is reached, we start our calculation with small m and enlarge it after the algorithm converged for the current m. This is also done in many DMRG programs. We enlarge the matrices A and increase their rank by filling the new rows and columns with small random numbers r, uniformly distributed in the interval $[-10^{-6}, 10^{-6}]$. The number of sweeps it takes until convergence is reached is similar to DMRG. For the present model, two or three sweeps are enough for each value of m.

Figure 2 shows that the algorithm indeed scales like m^3, and no longer like m^5. It is slightly faster on small systems, due to faster parts of the algorithm, and becomes slightly slower on large systems, likely due to memory access times. Our method (on a periodic system) requires a constant factor of about 10 as many operations per iteration as DMRG does on an open system, which is still very efficient.

Finally, we studied the convergence to the exact ground-state energy as a function of m. We investigated DMRG with obc and pbc, and the MPS algorithm with pbc, both the original version and our improved method. The relative error $\frac{\Delta E}{|E_0|}$ for these cases is plotted in Fig. 3. The relative error of the spin correlation function (not shown) is of similar magnitude with our improved method.

As has been well known, DMRG with obc performs much better than with pbc. With the MPS algorithm and pbc the relative error as a function of m is comparable to the error made with DMRG and obc. This has already been reported earlier.[8] The important point here is that the error remains the same when we introduce the approximations. Also, the number of sweeps until convergence is reached is similar for DMRG with obc and for MPS. We note that the convergence in Fig. 3 is consistent with exponential behavior in the spin 1 case and with a power law for spin $\frac{1}{2}$.

In a typical DMRG calculation, matrix dimensions $m \simeq 100-1000$ (and larger) are used. To illustrate the computational time scaling, suppose we study a model which requires $m=300$ states for obc with traditional DMRG. Then our approach gains a factor of roughly $m^5/m^3 \simeq 10^5$ over the method of Ref. 8, and even more over traditional DMRG.

In summary, by introducing a well-controlled approximate representation of products of MPS transfer matrices in terms of a singular value decomposition, we have formulated a circular MPS method for systems with periodic boundary conditions, which works with a computational effort comparable to that of DMRG with open boundary conditions.

We acknowledge support from the NSF under Grant No. DMR-0605444 (S.R.W.) and from NAWI Graz (P.P.).

[1] S. R. White, Phys. Rev. Lett. **69**, 2863 (1992).
[2] S. R. White, Phys. Rev. B **48**, 10345 (1993).
[3] U. Schollwöck, Rev. Mod. Phys. **77**, 259 (2005).
[4] S. Östlund and S. Rommer, Phys. Rev. Lett. **75**, 3537 (1995).
[5] S. Rommer and S. Östlund, Phys. Rev. B **55**, 2164 (1997).
[6] D. Porras, F. Verstraete, and J. I. Cirac, Phys. Rev. B **73**, 014410 (2006).
[7] G. Vidal, Phys. Rev. Lett. **91**, 147902 (2003).
[8] F. Verstraete, D. Porras, and J. I. Cirac, Phys. Rev. Lett. **93**, 227205 (2004).
[9] B. Paredes, F. Verstraete, and J. I. Cirac, Phys. Rev. Lett. **95**, 140501 (2005).
[10] F. Verstraete and J. Cirac, arXiv:cond-mat/0407066 (unpublished).
[11] G. Vidal, Phys. Rev. Lett. **98**, 070201 (2007).
[12] N. Schuch, M. M. Wolf, F. Verstraete, and J. I. Cirac, Phys. Rev. Lett. **100**, 040501 (2008).
[13] A. W. Sandvik and G. Vidal, Phys. Rev. Lett. **99**, 220602 (2007).
[14] D. Perez-Garcia, F. Verstraete, M. Wolf, and J. Cirac, Quantum Inf. Comput. **7**, 401 (2007).
[15] F. Verstraete (private communication).
[16] F. Verstraete *et al.* (unpublished).
[17] Denote $\mathbf{v}$ as an m^2 vector and V as the elements of $\mathbf{v}$ aligned as an $m \times m$ matrix. Then one can use the relation $(A \otimes B)\mathbf{v} = \mathrm{Vec}(BVA^T)$ to perform a matrix-vector product.
[18] A Lanczos approach would take an additional factor for convergence.