Applying matrix product operators to model systems with long-range interactions

Gregory M. Crosswhite[*]

Department of Physics, University of Washington, Seattle, Washington 98185, USA

A. C. Doherty and Guifré Vidal

School of Physical Sciences, University of Queensland, Brisbane, Queensland 4072, Australia

(Received 5 June 2008; published 14 July 2008)

An algorithm is presented which computes a translationally invariant matrix product state approximation of the ground state of an infinite one-dimensional (1D) system. It does this by embedding sites into an approximation of the infinite "environment" of the chain, allowing the sites to relax and then merging them with the environment in order to refine the approximation. By making use of matrix product operators, our approach is able to directly model any long-range interaction that can be systematically approximated by a series of decaying exponentials. We apply these techniques to compute the ground state of the Haldane-Shastry model [Phys. Rev. Lett. **60**, 635 (1988) and Phys. Rev. Lett. **60**, 639 (1988)] and present the results.

PACS number(s): 05.10.−a, 02.70.−c, 03.67.−a, 71.27.+a

I. INTRODUCTION

Typical macroscopic quantities of material contain particles numbering on the order of 10^{26}. The only hope of modeling such systems is to come up with a representation that is relatively small but nonetheless able to capture essential properties. For quantum systems this is particularly difficult because the state space grows exponentially with the size of the system, so there is a large amount of information which needs to be disposed.

One approach is to start with a small system and then build it up using a renormalization transformation at each step to project out "undesirable" states and keep the representation small. The key is to get one's criteria for "undesirable" correct. When studying low-energy behavior of a system, it turns out that the best selection criteria is not the energy (unless one is solving the Kondo problem[1]), but the weight in a density matrix, which incorporates interactions with a surrounding environment. This idea, first proposed by White[2] in his *density-matrix renormalization-group* (DMRG) algorithm, has proven very successful in building effective but compact representations of large one-dimensional (1D) systems. (See Ref. 3 for an excellent review of this subject.) However, it has not worked as well as desired in modeling systems with long-range interactions, and it does not produce manifestly translationally invariant representations of translationally invariant states. One can improve on both these fronts by working in momentum space rather than in real space,[4,5] but the computation is more costly as the transformation to momentum space introduces many additional operators that need to be renormalized at each step, and it can break symmetries that had been present in real space. Also, none of these methods are particularly effective in two or more dimensions.

An alternative approach is inspired by observing that the DMRG technique converges to a *matrix product state* (MPS) (Ref. 6). Rather than building up progressively larger systems, the iTEBD algorithm[7] assumes *a priori* that the system takes an infinite translationally invariant MPS form and then uses imaginary time evolution to converge to the ground state. This approach has the advantage that it obtains a very compact representation—in particular, it dispenses with the need to form renormalized versions of operators. Furthermore, it admits a natural extension to two dimensions[8] by using *projected entangled-pair states* (PEPS) (Ref. 9), a two-dimensional generalization of MPS. However, it is only able to handle systems with short-range interactions.

In this paper, we present an algorithm which also computes a translationally invariant MPS representation of a ground state but without being limited to short-range interactions. Instead of simulating an evolution in imaginary time as in the iTEBD algorithm, our approach follows the spirit of the variational technique described in Ref. 10 but enhanced to apply to infinite systems. Additionally, we build in the ability to work with arbitrary *matrix product operators*,[11,12] which makes this algorithm naturally suited to handle systems with long-range interactions.

The intuition behind our approach is as follows. Suppose that one were given an infinitely large translationally invariant 1D quantum chain held at zero temperature. If one were to add an additional site to this chain and allow the chain to relax, then one would expect that all of the old sites would remain unchanged, and the new site would change to match the rest. If one could emulate the environment experienced by a single site in this infinite chain, then by embedding a site into this environment and allowing the system to relax, one would obtain a site that "looks like" all of the sites in our infinite chain—giving us a compact representation for the chain.

We note that this algorithm bears some similarity to the product wave-function renormalization group (PWFRG) (Ref. 13) in that both algorithms have the same goal—to compute representations of ground states for infinite systems—and both use the same underlying matrix product structure to represent states.[14] The difference is that while the PWFRG approach seeks the infinite limit by starting with a small system and progressively enlarging it, we start from the very beginning with an *infinite* system represented in terms of effective environments, which we progressively refine. Furthermore, we incorporate matrix product operators into our approach which allow us to model systems with long-range interactions.

The remainder of this paper shall be organized as follows. First, we shall review the matrix product formalism and show how it allows us to construct a representation of a site embedded in and entangled with an environment. Second, we shall outline the operation of an algorithm, which uses an iteration procedure to approximate the effective environment of a site in an infinite chain; we shall then explain how one can obtain the expected values of operators from the output of this algorithm. Third, we shall explain how to obtain matrix product factorizations of Hamiltonians with exponentially decaying interactions. Finally, we shall pull all of these ideas together and show how they can be applied to obtain an accurate MPS representation of the ground state of the Haldane-Shastry model—an exactly solvable model with long-range interactions.

II. ALGORITHMS

A. Finding a MPS representation of the ground state

We start by recalling the form of a matrix product state for a finite system.[3] For a system with n sites, a matrix product state with boundaries takes the form

$$\mathcal{S}^{\alpha_1,\alpha_2,\ldots,\alpha_n} = \sum_{\{i_k\}} (L_0^{\mathcal{S}})_{i_1} (S_1)_{i_1 i_2}^{\alpha_1} (S_2)_{i_2 i_3}^{\alpha_2} \cdots (S_n)_{i_n i_{n+1}}^{\alpha_n} (R_{n+1}^{\mathcal{S}})_{i_{n+1}}.$$

The left-hand side corresponds to the rank-n tensor describing our state; each index α_k corresponds to a basis of a physical observable such as the z component of spin at site k. The rank-three tensors S_k are the site tensors, which have two kinds of indices: the superscript index, which has dimension d and corresponds to the actual physical observable; and the subscript indices, which have dimension χ and give information about entanglement between each site and its neighbors. The vectors L_0 and R_{n+1} give the boundary conditions.

The advantage of the matrix product representation is that it decomposes the quantum state into a collection of tensors; each of which is naturally associated with a site on the lattice. Because of this, we shall see that we can "zoom in" on one site and we only have to work with its corresponding tensor and a relatively small environment. In this respect, it is almost as nice as a truly local description that would let us ignore all the other sites entirely—a fact which is remarkable given that it can still capture nonlocal properties resulting from entanglement.

Given this form, we now consider how to compute the expected values of operators. As discussed in Refs. [11] and [12], operators of interest can typically be expressed as a matrix product operator,

$$\mathcal{O}^{(\alpha_1,\alpha_1'),(\alpha_2,\alpha_2'),\ldots,(\alpha_n,\alpha_n')}$$
$$= \sum_{\{i_k\}} (L_0^{\mathcal{O}})_{i_1} (O_1)_{i_1 i_2}^{\alpha_1,\alpha_1'} \cdots (O_n)_{i_n i_{n+1}}^{\alpha_n,\alpha_n'} (R_{n+1}^{\mathcal{O}})_{i_{n+1}}. \quad (1)$$

Having decomposed both the state and the operator into a product of tensors associated with lattice sites, we likewise decompose the expected value into a product of tensors by combining the state and operator tensor at each site to define,

$$(E_k^{\langle\mathcal{O}\rangle})_{i,j} \equiv (E_k^{\langle\mathcal{O}\rangle})_{(i',i'',i'''),(j',j'',j''')}$$
$$:= \sum_{\alpha_k,\alpha_k'} (S_k^*)_{i',j'}^{\alpha_k} (O_k)_{i'',j''}^{\alpha_k,\alpha_k'} (S_k)_{i''',j'''}^{\alpha_k'}.$$

(Note that we have taken all of the left and right subscript indices and grouped them together.) To complete our decomposition of the expected value, we shall also need to define the left and right boundaries,

$$(L_0^{\langle\mathcal{O}\rangle})_{i \equiv (i',i'',i''')} := (L_0^{\mathcal{S}^*})_{i'} (L_0^{\mathcal{O}})_{i''} (L_0^{\mathcal{S}})_{i'''},$$
$$(R_{n+1}^{\langle\mathcal{O}\rangle})_{j \equiv (j',j'',j''')} := (R_{n+1}^{\mathcal{S}^*})_{j'} (R_{n+1}^{\mathcal{O}})_{j''} (R_{n+1}^{\mathcal{S}})_{j'''}. \quad (2)$$

These boundary vectors can be thought of as defining an environment into which our system of n sites has been embedded. With these tensors so defined, the expected value of the matrix product operator $\mathcal{O}$ with respect to the matrix product state $\mathcal{S}$ is given by $\langle\mathcal{S}|\mathcal{O}|\mathcal{S}\rangle = L_0^{\langle\mathcal{O}\rangle} \cdot E_1^{\langle\mathcal{O}\rangle} \cdot E_2^{\langle\mathcal{O}\rangle} \cdots E_n^{\langle\mathcal{O}\rangle} \cdot R_{n+1}^{\langle\mathcal{O}\rangle}$, where $\cdot$ indicates summation over the adjoining subscript indices.

Now we have almost arrived at the picture described in Sec. I, except that we have an environment and multiple sites rather than a single site. To compute the effective environment of a particular site in this chain, we absorb all of the sites surrounding it into the system environment by contracting the $E_k^{\langle\mathcal{O}\rangle}$ matrices into the left and right boundaries—that is, we make the inductive definitions; $L_k^{\langle\mathcal{O}\rangle} := L_{k-1}^{\langle\mathcal{O}\rangle} \cdot E_k^{\langle\mathcal{O}\rangle}$ and $R_k := E_k^{\langle\mathcal{O}\rangle} \cdot R_{k+1}^{\langle\mathcal{O}\rangle}$. By doing this, we can write the expected value of the operator as a function of this site tensor,

$$\langle\mathcal{S}|\mathcal{O}|\mathcal{S}\rangle(S_k) \equiv S_k^* \circ M^{\langle\mathcal{O}\rangle_k} \circ S_k$$
$$:= \sum_{\substack{\alpha,\alpha',i',i''',\\ j',j'''}} (S_k^*)_{i',j'}^{\alpha} (M^{\langle\mathcal{O}\rangle_k})_{(i',j'),(i''',j''')}^{(\alpha),(\alpha')} (S_k)_{i''',j'''}^{\alpha'},$$

$$(3)$$

where $\circ$ denotes summation over the appropriate adjacent subscript and superscript indices, and

$$(M^{\langle\mathcal{O}\rangle_k})_{(i',j'),(i''',j''')}^{(\alpha),(\alpha')} := \sum_{i'',j''} (L_{k-1}^{\langle\mathcal{O}\rangle})_{i',i'',i'''} (O_k)_{i'',j''}^{\alpha_k,\alpha_k'} (R_{k+1}^{\langle\mathcal{O}\rangle})_{j',j'',j'''}.$$

We have now obtained an explicit means to compute the expected value of an observable $\mathcal{O}$ as a function of the site tensor at position k knowing only the environment of the site as given by $L_{k-1}^{\langle\mathcal{O}\rangle}$ and $R_{k+1}^{\langle\mathcal{O}\rangle}$. However, recall that we want to be more than passive observers—we want to actively move the system as close as possible to its ground state. Thus, we now want to vary the site tensor S_k in order to minimize the energy of the system. If we let the Hamiltonian (matrix product) operator be denoted by $\mathcal{H}$ and the identity operator by $\mathcal{I}$, then employing Eq. (3) we see that what we seek is the site tensor which minimizes the function

$$\mathcal{E}^k(S_k) := \frac{S_k^* \circ M^{\langle\mathcal{H}\rangle_k} \circ S_k}{S_k^* \circ M^{\langle\mathcal{I}\rangle_k} \circ S_k},$$

which gives us the (normalized) energy of the system as a function of only the site tensor at position k (i.e., assuming

that the environment has been frozen in place). Since this is a Rayleigh quotient, computing the minimizer is equivalent to solving a generalized eigenvalue problem.

There is a subtlety in this procedure, however, which is that one needs to take steps to make sure that the normalization matrix $M^{\langle \mathcal{I} \rangle_k}$ is well conditioned. This can be done by imposing the "right-normalization" condition $\sum_{\alpha,i}(S_l^*)_{ij}^{\alpha}(S_l)_{ij'}^{\alpha}=\delta_{jj'}$ on all the sites to the left of k, and the "left-normalization" condition $\sum_{\alpha,j}(S_l^*)_{ij}^{\alpha}(S_l)_{i'j}^{\alpha}=\delta_{ii'}$ on all the sites to the right of k. This ensures that the subscript indices connected to S_k are orthonormal, which makes $M^{\langle \mathcal{I} \rangle_k}$ well conditioned. We shall discuss how this is done in our algorithm shortly; for more information on how the normalization condition is used for finite-length systems, see Ref. 10.

Now we have all of the ingredients that we need to build our algorithm; a way of expressing a chain as a site tensor embedded in an environment and a way to relax a system in this representation constrained, so that only the site tensor (and not its environment) is changed. However, up to this point we have been working with a finite system; in order to apply these ideas to infinite systems, we shall modify our notation slightly to replace position labels with *iteration labels*. That is, we shall let $L_k^{\langle \mathcal{O} \rangle}$ and $R_k^{\langle \mathcal{O} \rangle}$ denote the infinite environment at iteration step k, and S_k denote the inserted site at this iteration. With this notation, the algorithm to find the ground state of a system is given in Table I. It is dominated by the costs of absorbing the site and operator tensors into the environment (in step 3biv and in the Lanczos iteration in step 3a), which are, respectively, $O(cd\chi^3)$ and $O(c^2d^2\chi^2)$ with c referring to the auxiliary dimension of the operator tensor.

We note here that this algorithm is unstable when applied directly to systems with antiferromagnetic interactions, because ground states of such systems are not invariant under translations of one site. Happily, since such systems are invariant under translations of *two* sites, there is a simple fix: work with blocks of two spins rather than one by setting $d=4$ and by multiplying two of the operator tensors together to form a two-site operator tensor;[16] this only affects the inputs to the algorithm and does not require changing the algorithm itself.

B. Computing expected values of operators

The output of the algorithm of Sec. II A is a normalized site tensor $\widetilde{S}$ which gives a translationally invariant representation for the ground state of the system. In order to make this useful, we need to have a way to obtain the expected value of operators from it. Of course, for extensive observables the expected value will be infinite since we have an infinitely large system; so instead, we seek the more useful quantity of the expected value per site. We shall consider two cases of operators: local operators and general matrix product operators. The basic trick in both cases is to note that $\lim_{N\to\infty}(E^{\langle \mathcal{O} \rangle})^N \cdot \vec{v} = \Lambda^N \cdot \vec{v}$, where Λ is a matrix in the maximal eigenspace of $E^{\langle \mathcal{O} \rangle}$—that is, in the infinite limit the action of the operator will converge to its action on the maximal eigenspace since its action on all other eigenspaces will be

TABLE I. Algorithm to compute a (normalized) translationally invariant matrix product state representation of the ground state of an infinite chain.

1. Set $\chi=1$ and $L^{\mathcal{S}}=R^{\mathcal{S}}=1$.
2. Compute $L_1^{\langle \mathcal{O} \rangle}$ and $R_1^{\langle \mathcal{O} \rangle}$ for the Hamiltonian (matrix product) operator $\mathcal{H}$ and the identity operator $\mathcal{I}$. (The latter, of course, has trivial boundaries.)
3. Until convergence has been reached:
   (a) Use an eigenvalue solver [such as ARPACK (Ref. 15)] to obtain the minimal eigenvalue and corresponding eigenvector of the generalized eigenvalue problem $M^{\langle \mathcal{H} \rangle_k}S_k=\lambda M^{\langle \mathcal{I} \rangle_k}S_k$. To accelerate convergence, feed in S_{k-1} as a starting estimate for the eigenvector.
   (b) If this is an odd-numbered step, then right normalize S_k and contract into the left boundary. Specifically:
      (i) Merge the superscript and first subscript index of S_k to form a matrix and compute the singular value decomposition (SVD), $U \cdot \Sigma \cdot V^\dagger$.
      (ii) Set $\widetilde{S}_k := U \cdot V^\dagger$ and ungroup indices to return $\widetilde{S}_k$ to its original rank-three shape.
      (iii) Compute $E_k^{\langle \mathcal{O} \rangle}$ for $\mathcal{H}$ and $\mathcal{I}$ using the normalized $\widetilde{S}_k$.
      (iv) Contract the site into the left boundary by setting $L_{k+1}^{\langle \mathcal{O} \rangle} := L_k^{\langle \mathcal{O} \rangle} \cdot E_k^{\langle \mathcal{O} \rangle}$ and $R_{k+1}^{\langle \mathcal{O} \rangle} := R_k^{\langle \mathcal{O} \rangle}$. (This step "absorbs" the site into the environment.)
   (c) If this is an even-numbered step, perform an analogous process, but *left* normalize S_k by merging the superscript and *second* index together before the SVD and then contract the normalized site into the *right* boundary.
4. If a better approximation to the ground state is desired, then increase χ for the system and repeat step 3. Increasing the dimension of a subscript can be done without altering the state by multiplying the adjoining tensors by a $\chi \times (\chi + \Delta \chi)$ matrix and its inverse; this allows one to build on the work of previous iterations (and hence accelerate convergence) rather than having to start from scratch with the new χ.

negligible by comparison. (Assuming, of course, that $\Lambda \cdot \vec{v} \neq 0$.)

First we consider local operators, which can be expressed as sum of terms of the form $I^{\otimes\infty} \otimes O_1 \otimes O_2 \otimes \ldots \otimes O_N \otimes I^{\otimes\infty}$ such as a magnetic-field operator, $I^{\otimes\infty} \otimes Z \otimes I^{\otimes\infty}$ or a two-point correlator $I^{\otimes\infty} \otimes Z \otimes I^{\otimes r} \otimes Z \otimes I^{\otimes\infty}$. Since our system is translationally invariant to compute the expected value per site of this operator, we need only to evaluate the expected value of one term in the sum. Assuming that $E^{\langle \mathcal{I} \rangle}$ has a nondegenerate maximal eigenvalue, we have that for (almost) any vector $\vec{v}$, $(E^{\langle \mathcal{I} \rangle})^\infty \cdot \vec{v} \sim \vec{v}^R$ and $\vec{v} \cdot (E^{\langle \mathcal{I} \rangle})^\infty \sim \vec{v}^L$, where $\vec{v}^L$ and $\vec{v}^R$ are the respective left and right eigenvectors corresponding to the maximal eigenvalue. Ergo, the expected value of one term of this operator (and thus the expected value per site) is given by

$$\frac{\vec{v}^L \cdot E^{O_1} \cdot E^{O_2} \ldots E^{O_N} \cdot \vec{v}^R}{\vec{v}^L \cdot (E^{\langle \mathcal{I} \rangle})^N \cdot \vec{v}^R}.$$

Next we consider general matrix product operators. We start by writing an expression for the expected value per site for a finite chain of length N whose site tensors are copies of the (normalized) site tensor obtained from the variational algorithm in Sec. II A. Since the infinite chain has no boundaries to compute the expected value of the finite chain, we need to explicitly supply left and right boundaries, respectively, L^S and R^S; and so the expected value per site of $\mathcal{O}$ is a function of the boundaries given by

$$\mathcal{E}^N(L^S, R^S) := \frac{1}{N} \frac{L^{\langle \mathcal{O} \rangle} \cdot (E^{\langle \mathcal{O} \rangle})^N \cdot R^{\langle \mathcal{O} \rangle}}{L^{\langle \mathcal{I} \rangle} \cdot (E^{\langle \mathcal{I} \rangle})^N \cdot R^{\langle \mathcal{I} \rangle}},$$

where $L^{\langle \mathcal{O} \rangle}$, $R^{\langle \mathcal{O} \rangle}$, $L^{\langle \mathcal{I} \rangle}$, and $R^{\langle \mathcal{I} \rangle}$ are implicitly functions of L^S and R^S given by Eq. (2). Assuming $\mathcal{O}$ corresponds to an extensive observable, we expect that $\lim_{N \to \infty} \mathcal{E}^N(L_S, R_S) = \langle \mathcal{O} \rangle$; that is, in the infinite limit the expected value per site converges to some number $\langle \mathcal{O} \rangle$, which is independent of the boundaries. With this physically reasonably assumption, we can reason out about the structure of $E^{\langle \mathcal{O} \rangle}$ and $E^{\langle \mathcal{I} \rangle}$ to obtain an algorithm for computing $\langle \mathcal{O} \rangle$.

We first observe that since the maximal eigenvalue of $E^{\langle \mathcal{I} \rangle}$ is one (due to the normalization of the site tensor $\tilde{S}$), so must be the maximal eigenvalue of $E^{\langle \mathcal{O} \rangle}$, since otherwise $\mathcal{E}^N$, would be exponential in N. Furthermore, since $\mathcal{E}^N$ is linear in N for large N, the maximal eigenspace of $E^{\langle \mathcal{I} \rangle}$ must have a Jordan block structure–that is, there must be a matrix $U \equiv [\vec{u}_1 \vec{u}_2]$ with orthonormal columns $\vec{u}_i$ that provide a basis for this eigenspace such that $A := U \cdot E^{\langle \mathcal{O} \rangle} \cdot U^\dagger = \begin{bmatrix} 1 & \alpha \\ 0 & 1 \end{bmatrix}$. The matrix element α can be thought of as giving us the unnormalized expected value of $\mathcal{O}$ per site. In order to normalize it, we observe that as $N \to \infty$,

$$\mathcal{E}^N \to \frac{1}{N} \frac{L^{\langle \mathcal{O} \rangle} \cdot (\vec{u}_1 \vec{u}_1^\dagger + \vec{u}_2 \vec{u}_2^\dagger + N \alpha \vec{u}_1 \vec{u}_2^\dagger) \cdot R^{\langle \mathcal{O} \rangle}}{L^{\langle \mathcal{I} \rangle} \cdot (E^{\langle \mathcal{I} \rangle})^N \cdot R^{\langle \mathcal{I} \rangle}}$$

$$\to \alpha \frac{L^{\langle \mathcal{O} \rangle} \cdot (\vec{u}_1 \vec{u}_2^\dagger) \cdot R^{\langle \mathcal{O} \rangle}}{L^{\langle \mathcal{I} \rangle} \cdot (E^{\langle \mathcal{I} \rangle})^N \cdot R^{\langle \mathcal{I} \rangle}}.$$

Since we expect $\mathcal{E}^N$ to be independent of the boundaries for large N, it must be the case that

$$L^{\langle \mathcal{I} \rangle} \cdot (E^{\langle \mathcal{I} \rangle})^N \cdot R^{\langle \mathcal{I} \rangle} \to \beta [L^{\langle \mathcal{O} \rangle} \cdot (\vec{u}_1 \vec{u}_2^\dagger) \cdot R^{\langle \mathcal{O} \rangle}]$$

$$\equiv \beta [L^{\langle \mathcal{I} \rangle} \cdot (\vec{v}_1^L \vec{v}_2^{R\dagger}) \cdot R^{\langle \mathcal{I} \rangle}],$$

where $\vec{v}_k^L := \vec{u}_k \cdot L^{\langle \mathcal{O} \rangle} \equiv \Sigma_{i''} u_{k,(i',i'',i''')} L_{i''}^{\mathcal{O}}$ and $\vec{v}_k^R := \vec{u}_k \cdot R^{\mathcal{O}} \equiv \Sigma_{i''} u_{k,(i',i'',i''')} R_{i''}^{\mathcal{O}}$; that is, $\vec{v}_k^L$ and $\vec{v}_k^R$ are the projections of the eigenspace of $E^{\langle \mathcal{O} \rangle}$ into a space applicable to $E^{\langle \mathcal{I} \rangle}$ by dotting out the respective left and right boundaries of the matrix product operator $\mathcal{O}$. As put in another way, if we let $V^L := [\vec{v}_1^L \vec{v}_2^L]$ and $V^R := [\vec{v}_1^R \vec{v}_2^R]$, then the limiting action of $E^{\langle \mathcal{I} \rangle}$ in the (projected) eigenspace is given by $B := V^L \cdot E^{\langle \mathcal{I} \rangle} \cdot V^{R\dagger} = \begin{bmatrix} 0 & \beta \\ 0 & 0 \end{bmatrix}$. The matrix element β gives us the normalization constant, so that $\langle \mathcal{O} \rangle := \lim_{N \to \infty} \mathcal{E}^N = \alpha / \beta$.

It remains to extract α and β from the respective matrices A and B. Our eigenvalue solver is not guaranteed to give us a basis of the eigenspace that makes A and B have the nice

TABLE II. Algorithm to compute $\langle \mathcal{O} \rangle$ for general matrix product operator $\mathcal{O}$.

1.	Compute the maximal two-dimensional eigenspace of $E^{\langle \mathcal{O} \rangle}$; for example, by using ARPACK (Ref. 15) and requesting a Schur basis rather than a (nonexistent) eigenvector basis; and store the basis vectors in the columns of a matrix U.
2.	Express the action of $E^{\langle \mathcal{O} \rangle}$ in this space by computing the 2×2 matrix $A := U \cdot E^{\langle \mathcal{O} \rangle} \cdot U^\dagger$.
3.	Extract the unnormalized expected value from this matrix by computing $\alpha := \sqrt{\mathrm{Tr}(A^\dagger \cdot A) - 2}$.
4.	Project out the operator-dependent boundary conditions from the vectors in this eigenspace to obtain $V^L := L^{\mathcal{O}} \cdot U \equiv \Sigma L_{i''}^{\mathcal{O}} U_{(i',i'',i''')_,k}$ and $V^R := R^{\mathcal{O}} \cdot U \equiv \Sigma R_{i''}^{\mathcal{O}} U_{(i',i'',i''')_,k}$.
5.	Express the action of $E^{\langle \mathcal{I} \rangle}$ in this projected space by computing the 2×2 matrix $B := V^L \cdot E^{\langle \mathcal{I} \rangle} \cdot V^{R\dagger}$.
6.	Extract the normalization constant by computing $\beta := \sqrt{\mathrm{Tr}(B^\dagger \cdot B)}$.
7.	Obtain the (normalized) expected value by computing $\langle \mathcal{O} \rangle := \alpha / \beta$.

form above, but happily it is easy to see that $\alpha = \sqrt{\mathrm{Tr}(A \cdot A^\dagger) - 2}$ and $\beta = \sqrt{\mathrm{Tr}(B \cdot B^\dagger)}$, and these formulas are invariant under similarity transforms of A and B, so they work independent of the basis we are given. The algorithm we have derived in the preceding paragraphs is summarized in Table II.

III. MATRIX PRODUCT FACTORIZATION OF EXPONENTIALLY DECAYING INTERACTIONS

In the algorithms of Sec. II, we assume that there is a matrix product representation of our Hamiltonian. It has previously been shown that there are matrix product factorizations of Hamiltonians with short-range interactions.[11,12] In this section, we extend these results to show that there are also matrix product factorizations of Hamiltonians with long-range exponentially decaying interactions as well.

So consider a Hamiltonian of an N-site system experiencing a sum of long-range exponentially decaying interactions,

$$\mathcal{H} := \sum_{i+1+r+1+j=N} \left(\sum_n \alpha_n \beta_n^r \right) \mathbf{I}^{\otimes i} \otimes \mathbf{X} \otimes \mathbf{I}^{\otimes r} \otimes \mathbf{X} \otimes \mathbf{I}^{\otimes j},$$

where $\mathbf{I}$ is the identity operator and $\mathbf{X}$ is the Pauli operator σ^X. We shall now employ the formalism in Ref. 12 to obtain a factorization of this Hamiltonian. We start by changing our interpretation of the Hamiltonian—rather than thinking of it as a sum of tensor product terms with complex coefficients, we shall instead think of it as a function which maps arbitrary strings of X and I symbols to complex numbers. In our sum each term takes the form $(\Sigma_n \alpha_n \beta_n^r) \mathbf{I}^{\otimes i} \otimes \mathbf{X} \otimes \mathbf{I}^{\otimes r} \otimes \mathbf{X} \otimes \mathbf{I}^{\otimes j}$ with $i, j, r \geq 0$, so our corresponding function maps strings of the form $\mathtt{I}^i \mathtt{X} \mathtt{I}^r \mathtt{X} \mathtt{I}^j$ to $\Sigma_n \alpha_n \beta_n^r$ and all other strings to zero.

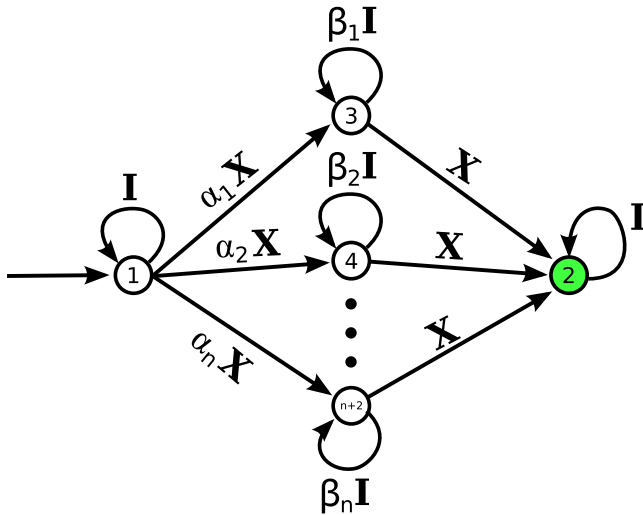We shall now construct a *finite state automaton*, which

FIG. 1. (Color online) Finite state automata representation of a sum of exponentially decaying *XX* interactions.

computes this function. A finite state automaton can be thought of as a machine which reads through an input string and changes its state in response to each input symbol according to a set of transition rules. The transitions are non-deterministic and weighted—that is, the automaton can take many transitions simultaneously, and at the end of the string it outputs a number corresponding to the sum of the product of the weights along each sequence of transitions that it took. The automaton starts on a designated *initial* state; if it does not end on a designated *accept* state or it encounters a symbol without an associated transition, then it outputs a zero (overriding other weights).

The automaton which computes the function representing our operator is illustrated in Fig. 1. States are indicated by circles, and transition rules are indicated by arrows labeled with a symbol and a weight (one if not otherwise specified). An unconnected arrow designates the initial state (1) and shading designates the accept state (2). A good way to think about what is going on is that terms are generated by each possible walk from state (1) to state (2). So for example, by taking the path $1 \rightarrow 1 \rightarrow 3 \rightarrow 3 \rightarrow 3 \rightarrow 2$ we generate the term $(\alpha_1 \beta_1^2) \mathbf{I} \otimes \mathbf{X} \otimes \mathbf{I} \otimes \mathbf{I} \otimes \mathbf{X}$; summing over all walks for strings of the form $\texttt{IXI}^2\texttt{X}$ we obtain the desired coefficient $\Sigma_n \alpha_n \beta_n^2$.

From this automaton, we immediately obtain a matrix product operator representation. The initial and accept states give us the values for the left and right boundaries: $(L^{\mathcal{O}})_k = \delta_{1,k}$ and $(R^{\mathcal{O}})_k = \delta_{2,k}$. The elements of the operator tensor $O_{i,j}^{\mathbf{A}}$ are given by the weight on the $i \rightarrow j$ transition with the symbol corresponding to operator $\mathbf{A}$ (zero if no such transition exists); for example, we have that $O_{1,3}^{\mathbf{X}} = \alpha_1$. To get a feeling for why this works, observe that a run of the automaton is equivalent to starting with a vector giving initial weights on the states, multiplying this vector some number of times by a transition matrix and then dotting the result with a vector that filters out all but the weights on the accept states; this procedure is exactly equivalent to the form of Eq. (1).

This process is easily extended to include terms with arbitrary spin coupling interactions such as $\sigma^X \sigma^Y$, $\sigma^Y \sigma^Y$, $\sigma^Z \sigma^Z$,

etc. Furthermore, one can combine a sum of several such interactions into a single automaton by having them all share the same starting and ending states.

IV. RESULTS: HALDANE-SHASTRY MODEL

Now we pull all of the ideas from Secs. I–III together and apply them to tackle the Haldane-Shatry model.[17,18] In the infinite limit this model is given by the Hamiltonian $\mathcal{H} = \Sigma_i \Sigma_r \vec{\sigma}_i \cdot \vec{\sigma}_{i+r} / r^2$, which features an antiferromagnetic dipole interaction which falls off with the square of the distance between sites. (Note that since this model features antiferromagnetic interactions, we need to work with blocks of two sites as discussed at the end of Sec. II A.) Although $\mathcal{H}$ cannot be expressed exactly as a matrix product operator, we can approximate it arbitrarily well by a sum of exponentially decaying interactions $\mathcal{H} \approx \Sigma_i \Sigma_r \vec{\sigma}_i \cdot \vec{\sigma}_{i+r} (\Sigma_n \alpha_n \beta_n^{r-1})$ (with $\alpha_n \in \mathbb{R}$ and $|\beta_n| \leq 1$), which can be factored exactly using the technique in Sec. III. Since there are three spin-coupling interactions, $\sigma_i^X \sigma_{i+r}^X$, $\sigma_i^Y \sigma_{i+r}^Y$, and $\sigma_i^Z \sigma_{i+r}^Z$, which we combine into a single automaton as discussed at the end of Sec III, we obtain an automaton with a number of states equal to three times the number of terms in the expansion N plus two more for the starting and ending states; this quantity gives us the size of the auxiliary dimension for the corresponding matrix product operator $c = 3N + 2$.

It remains to find the coefficients in this expansion. One approach is to numerically solve for the coefficients which minimize the sum of the squares of the difference between the approximation and the exact potential for distances up to some cutoff—that is, to find the minimizer of the function

$$f(\alpha_1, \beta_1, \ldots, \alpha_N, \beta_N) = \sum_{i=1}^{N} \sum_{r=1}^{r_{\text{cutoff}}} \left(\alpha_i \beta_i^{r-1} - \frac{1}{r^2} \right)^2,$$

where r_{cutoff} should be chosen to be just beyond the maximum effective range of the approximation, since larger values of r_{cutoff} result in a longer running time for the minimization without resulting in a better fit.

For our application of the algorithm, we used a nonlinear least-squares minimization routine from MINPACK to find coefficients for expansions with three, six, and nine terms. The resulting approximate potentials are plotted along side the exact potential in Fig. 2. The upper cutoff on r was set to 10000, because as can be seen in Fig. 2 this was just beyond the effective maximum range obtainable from a nine-term approximation.

Given this approximate matrix product factorization, we applied the algorithm in Table I to compute a translationally invariant matrix product state representation of the ground state for selected values of χ, employing each of the three-, six-, and nine-term expansions. The energy per site was computed using the algorithm in Table II and compared to the exact value obtained from Ref. 18. The difference between these values (i.e., the residual) is plotted for each expansion as a function of χ in Fig. 3. Note that the residuals for all three expansions agree up to some point and then diverge to different "floors." This is because at first the small value of χ is the dominating factor which limits the fidelity of the
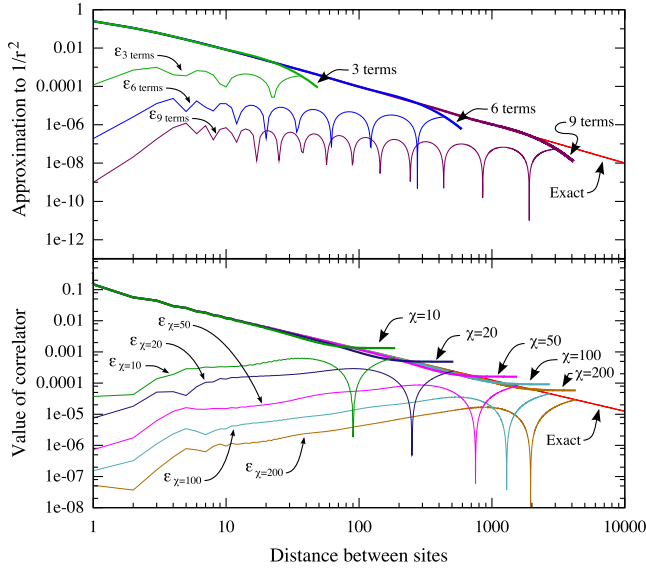
FIG. 2. (Color online) (Top) Decaying three-, six-, and nine-term exponential approximations to $1/r^2$ potential. (Bottom) Two-point correlator for selected values of χ using the nine-term exponential approximation; a residual of the correlator for each value of χ is plotted simultaneously and labeled by ε_χ.

ground state and then later as χ becomes large, the finite number of terms in the exponential approximation becomes the dominating factor.

For the nine-term expansion, we also computed the two-point correlator—that is, $C(r) := \langle \sigma^X \otimes I^{\otimes(r-1)} \otimes \sigma^X \rangle$, using the algorithm given in Sec. II B for computing the expected value of a local operator. The result for several values of χ is plotted in Fig. 2 against the exact value from Ref. 18. Note that our approximation gets good agreement up to some length; after which, it becomes a constant. This is because the algorithm is attempting to approximate this correlator using a sum of decaying exponentials plus a constant term analogous to how we used a sum of decaying exponentials to approximate the $1/r^2$ interactions. By increasing χ, we are increasing the number of terms available to track the correlator, which results in systematic improvement.

V. CONCLUSIONS

To summarize, in this paper we have presented an algorithm for computing the ground state of infinite 1D systems. This algorithm differs from the iTEBD algorithm[7] in that it uses a variational approach instead of imaginary time evolu-
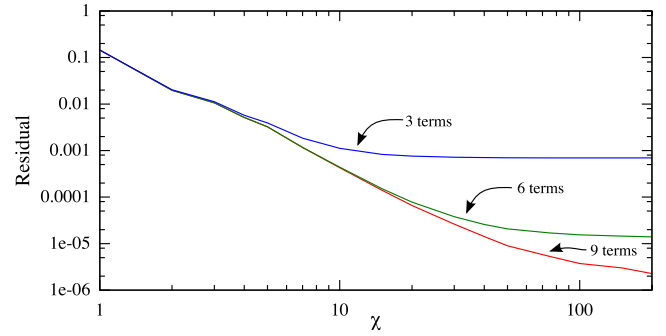

FIG. 3. (Color online) Difference between the energy of the computed state and the energy of the exact ground state plotted for each of the three exponential expansions that were employed.

tion and from the PWFRG (Ref. 13) in that it considers an infinite system from the start. Furthermore, since the algorithm itself employs matrix product operators, it has the important advantage of being capable of modeling long-range interactions and in particular, to any interaction which can be approximated by a sum of decaying exponentials. In order to benchmark the algorithm, we have computed an infinite MPS for the ground state of the Haldane-Shastry model. The corresponding two-point correlators are in remarkable agreement with the exact solution up to distances above a thousand spins.

In conclusion, our results indicate that this algorithm adds significantly to the existent tools to address 1D many-body systems since it allows the properties of bulk-scale materials to be studied for realistic long-range potentials. Furthermore, it admits a natural extension to lattice systems in higher spatial dimensions for which work is currently in progress.

We note that upon completion of this work, we learned of simultaneous work on an equivalent algorithm by McColloch in Ref. 19. His presentation includes a detailed comparison of the convergence of the iTEBD (Ref. 7) and the variational approaches.

*gcross@phys.washington.edu

[1] K. G. Wilson, Rev. Mod. Phys. **47**, 773 (1975).

[2] S. R. White and R. M. Noack, Phys. Rev. Lett. **68**, 3487 (1992).

[3] U. Schollwöck, Rev. Mod. Phys. **77**, 259 (2005).

[4] T. Xiang, Phys. Rev. B **53**, R10445 (1996).

[5] S. Nishimoto, E. Jeckelmann, F. Gebhard, and R. M. Noack,

Phys. Rev. B **65**, 165114 (2002).

[6] S. Östlund and S. Rommer, Phys. Rev. Lett. **75**, 3537 (1995); S. Rommer and S. Östlund, Phys. Rev. B **55**, 2164 (1997).

[7] G. Vidal, Phys. Rev. Lett. **98**, 070201 (2007).

[8] J. Jordan, R. Orus, G. Vidal, F. Verstraete, and J. I. Cirac, arXiv:cond-mat/0703788 (unpublished).

[9] F. Verstraete and J. I. Cirac, arXiv:cond-mat/0407066 (unpublished).

[10] F. Verstraete, D. Porras, and J. I. Cirac, Phys. Rev. Lett. **93**, 227205 (2004).

[11] I. P. McCulloch, J. Stat. Mech.: Theory Exp. (2007) P10014.

[12] G. M. Crosswhite and D. Bacon, arXiv:0708.1221, Phys. Rev. A (to be published).

[13] T. Nishino and K. Okunishi, J. Phys. Soc. Jpn. **64**, 4084 (1995); Y. Hieida, K. Okunishi, and Y. Akutsu, Phys. Lett. A **233**, 464 (1997).

[14] K. Ueda, T. Nishino, K. Okunishi, Y. Hieida, R. Derian, and A. Gendiar, J. Phys. Soc. Jpn. **75**, 014003 (2006).

[15] R. Lehoucq, D. Sorensen, and C. Yang, ARPACK users' guide, 1997 (http://www.caam.rice.edu/software/ARPACK).

[16] Note that this approach does not scale well beyond two sites; that is, given an interaction which is symmetric under translations of n sites, blocking the sites together grows the size of the representation by a factor of 2^n. An alternative strategy is to add n sites at a time to the center of the system and then use a sweeping algorithm like that described in Ref. 10 to optimize the site tensors; the final representation is then given by n site tensors rather than one.

[17] F. D. M. Haldane, Phys. Rev. Lett. **60**, 635 (1988).

[18] B. S. Shastry, Phys. Rev. Lett. **60**, 639 (1988).

[19] I. P. McCulloch, arXiv:0804.2509 (unpublished).