Introduction to basic density-matrix renormalization group calculations

Bo Xiao[1, 2, *]

[1] *Materials Science and Technology Division, Oak Ridge National Laboratory, Oak Ridge, Tennessee 37831, USA*
[2] *Quantum Science Center. Oak Ridge, Tennessee 37831, USA*

(Dated: July 15, 2024)

I. INTRODUCTION

In this tutorial, we will introduce a powerful numerical method, *density matrix renormalization group* (DMRG), and demonstrate how to simulate some paradigmatic one-dimensional (1D) spin models using this approach. Based on renormalization group ideas, the DMRG algorithm was introduced to calculate low-lying states and their observables in 1D lattice models [1–3]. It is realized later that DMRG is closely linked to a special quantum state class, so-called *matrix product state* (MPS) [4–6], or equivalently, what some mathematicians call a tensor train [7]. In this tutorial, we will set up the numerical algorithm using the MPS language. Within this frame, DMRG can be understood as a variational optimization algorithm over MPS.

It is true that MPS are particularly well suited for describing gapped 1D quantum lattice systems with local interactions [6, 8–11]. Different tensor network structures have been developed for other scenarios: projected entangled pair states [12] for tackling two-dimensional systems, multiscale entanglement renormalization *ansätz* (MERA) [13] for critical systems, and tree tensor network etc. In addition, numerical methods such as time-evolving block decimation (TEBD) [14, 15] and time-dependent variational principle (TDVP) [16, 17] enables the computation of time evolution of an MPS.

A. Matrix Product States

Let us set up a quantum system that lives on a 1D chain with N sites. The dimension of the local Hilbert space of states $\{\sigma_i\}$ is set to be d. For example, for an interacting spin-$\frac{1}{2}$ model, the local states are $|\uparrow\rangle$, $|\downarrow\rangle$ and $d = 2$. Pure states are then defined on the d^N-dimensional Hilbert space

$$\mathcal{H} = \bigotimes_{\ell=1}^{N} \mathcal{H}_\ell, \tag{1}$$

and the most general state can be expanded in a computational basis as

$$|\Psi\rangle = \sum_{s_1,s_2,\ldots,s_N} c_{s_1,s_2,\ldots,s_N} |s_1, s_2, \ldots, s_N\rangle, \tag{2}$$

where s_ℓ denote the quantum numbers associated with the basis states $|s_\ell\rangle \in \mathcal{H}_\ell$. It is straightforward to see the total Hilbert space dimension is $\dim(\mathcal{H}) = d^N$. MPS can be understood as the simplest tensor network structure that compress the wave function to reduce the exponential dependence of the number of required parameters. The core idea of MPS is to compress the many-body wave function by decomposing the wave function coefficients into a product of matrices,

$$c_{s_1,s_2,\ldots,s_N} = \sum_{a_1,a_2,\ldots,a_N} A^{s_1}_{a_1} A^{s_2}_{a_1,a_2} \ldots A^{s_N}_{a_{N-1}}. \tag{3}$$

After this compression, the number of parameters needed to parameterize the MPS is on the order of $\mathcal{O}(Nd\chi^2)$, where χ is the bond dimension determined by the entanglement of bipartition.

* xiaob@ornl.gov

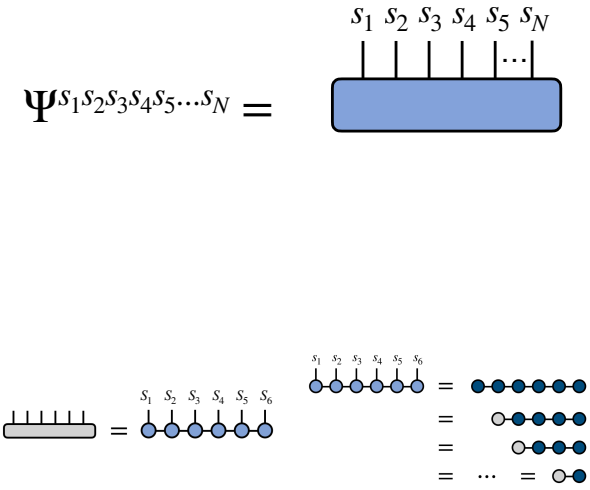


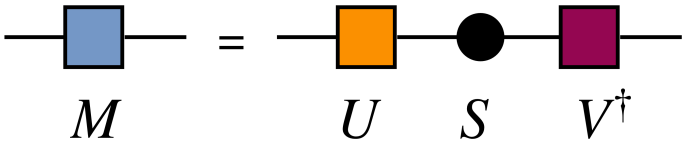
FIG. 1: Using a six-site spin chain as a toy model to show the compression of wave function coefficients, represented by the huge blue tensors, into matrix product states. When the local states s_ℓ are fixed on each site, tracing back the coefficient for that specific configuration becomes matrix multiplication in the MPS form.

B. Singular Value Decomposition

In this tutorial, it is worthwhile to introduce the singular value decomposition (SVD). This numerical method for matrix decomposition is crucial for tracing out "unimportant" degrees of freedom, thereby enabling the approximation of the real ground state in a much smaller Hilbert space. For an arbitrary $(m \times n)$ matrix M, there exists the singular value decomposition with $\ell = \min(m, n)$

$$M = USV^\dagger \tag{4}$$

where U is $m \times \ell$ matrix with orthonormal columns ($U^\dagger U = I$), S is a diagonal $\ell \times \ell$ matrix with non-negative entries, and $V^\dagger$ is a $\ell \times n$ matrix with orthonormal rows ($V^\dagger V = I$).



Furthermore, we want to mention that QR decomposition, another matrix decomposition method, is much faster and also heavily used in the MPS-based algorithm.

C. Reformulate Tensor Networks in The Matrix Product States Language

For any given wave function, applying the SVD in an iterative manner allows to bring a tensor into MPS form. Let's show the first two steps of the procedure to illustrate how this idea works. Starting from the original d^N dimensional vector of wave function coefficients $c_{s_1,s_2,\ldots,s_N}$, we can reshape the coefficients into a matrix form $C_{s_1,(s_2,\ldots,s_N)} = c_{s_1,s_2,\ldots,s_N}$ of dimensions $d \times d^{N-1}$. Here d is the dimension of the local Hilbert space and N is the number of physical sites. In other words, we treat s_1 as the row index and the composite index $(s_2, s_3, \ldots, s_N)$ as the column index. Then we apply a SVD to the matrix

$$
\begin{aligned}
c_{s_1,s_2,\ldots,s_N} &= C_{s_1,(s_2,\ldots,s_N)} \\
&= \sum_{a_1=1}^{\chi_1} U_{s_1,a_1} S_{a_1,a_1} (V^\dagger)_{a_1,(s_2,\ldots,s_N)} \equiv \sum_{a_1=1}^{\chi_1} U_{s_1,a_1} c_{a_1,s_2,\ldots,s_N},
\end{aligned} \tag{5}
$$

where we defined $c_{a_1,s_2,\ldots,s_N} = S_{a_1,a_1}(V^\dagger)_{a_1,(s_2,\ldots,s_N)}$. Let's continue decomposing $c_{a_1,s_2,\ldots,s_N}$,

$$
\begin{aligned}
c_{a_1,s_2,\ldots,s_N} &= C_{(a_1,s_2),(s_3,\ldots,s_N)} \\
&= \sum_{a_2=1}^{\chi_2} U_{(a_1,s_2),a_2} S_{a_2,a_2} (V^\dagger)_{a_2,(s_3,\ldots,s_N)} \equiv \sum_{a_2=1}^{\chi_2} U_{(a_1,s_2),a_2} c_{a_2,s_3,\ldots,s_N}.
\end{aligned} \tag{6}
$$

By identifying $A^{s_1}_{a_1} \equiv U_{s_1,a_1}$ and $A^{s_2}_{a_1,a_2} = U_{(a_1,s_2),a_2}$, we can rewrite the original wave function coefficient as

$$
c_{s_1,s_2,\ldots,s_N} = \sum_{a_1,a_2} A^{s_1}_{a_1} A^{s_2}_{a_1,a_2} c_{a_2,s_3,\ldots,s_N}. \tag{7}
$$

By applying the SVD iteratively, we can decompose the wave function coefficients into a product of matrices

$$
c_{s_1,s_2,\ldots,s_N} = \sum_{a_1,a_2,\ldots,a_N} A^{s_1}_{a_1} A^{s_2}_{a_1,a_2} \ldots A^{s_N}_{a_{N-1}}. \tag{8}
$$

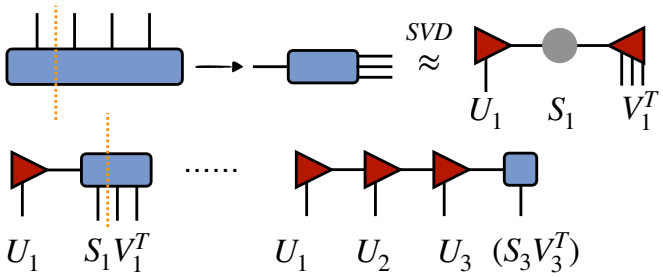


FIG. 2: Schematic illustration of decomposing a rank-4 tensor into a product of matrices using a tensor diagram. Applying the SVD decomposition iteratively.

Given that the scope of this tutorial note is to introduce the computation of ground states in spin models using DMRG, there are many topics we will not be able to cover. Indeed, we haven't introduced a compression scheme in the above procedure. Under specific relevant conditions, the maximum rank can be significantly reduced, which is associated with the truncation error ϵ. Furthermore, the matrix product state does not change under the insertion of $XX^{-1} = I$ between matrices A^{s_i} and $A^{s_{i+1}}$, which implies a gauge transformation $A^{s_i} \to A^{s_i}X$ and $A^{s_{i+1}} \to X^{-1}A^{s_{i+1}}$. Exploiting this gauge freedom is beneficial in manipulating MPS. In the tutorial session, we will delve deeper into the topics of compression scheme, canonical forms, and matrix product operators. For readers seeking more technical details, Ref. [6, 8, 9] offer further insights into these topics.

II. HANDS-ON CODING

A. Installing Julia and the Julia version of ITensor

In this tutorial, we will be using the Julia programming language along with the Julia version of ITensor [18, 19]. It is strongly recommended to download and install Julia before the classes. You can find platform-specific

installation instruction at https://julialang.org/downloads/. Additionally, it is highly recommended to install the Julia version of ITensor through the Julia package manager. A detailed introduction to the ITensor package is available at https://www.scipost.org/SciPostPhysCodeb.4.

```
# Installing the Julia version of ITensor and
# Installing HDF5 (stands for Hieerarchical Data Format v5), which a file format for storing and managing data
$ julia
julia> ]
pkg> add ITensors
pkg> add HDF5
```

B. The XXZ Model

The spin-$\frac{1}{2}$ XXZ model is a deformation of the Heisenberg model which breaks the SU(2) symmetry down to a U(1) symmetry. The XXZ chain is defined by the Hamiltonian

$$H = \sum_{i=1}^{N} \left(S_i^x S_{i+1}^x + S_i^y S_{i+1}^y + \Delta S_i^z S_{i+1}^z \right). \tag{9}$$

Δ is the anisotropy parameter, where $\Delta = 1$ is the isotropic case. The system is in a quasi-long-ranged ordered nonmagnetic gapless phase when Δ satisfies $-1 < \Delta \leq 1$. When $\Delta > 1$, the ground state possesses a long-range antiferromagnetic order in which the spin correlation functions decay exponentially.

Before delving into the actual coding for the XXZ model, let's discuss the importance of selecting an appropriate boundary condition for numerical simulations on a finite lattice. In the conventional DMRG, optimization is performed over matrix product states (MPS) with open boundary condition (OBC). The computational cost of this approach is on the order of $\mathcal{O}(Nd\chi^2)$, where N represents the number of sites on a one-dimensional chain, d denotes the dimension of the local Hilbert space, and χ is the bond dimension. In sharp contrast, using a periodic boundary condition (PBC) changes the scaling to $\mathcal{O}(Nd\chi^3)$ to reach the same accuracy. Furthermore, we want to mention that the boundary effects can be rigorously removed by using infinite boundary condition through infinite DMRG [20, 21] or variational uniform matrix product states (VUMPS) [22], if the calculation has converged.

```
# Conduct a basic DMRG calculation on the one-dimensional XXZ model
using ITensors
using HDF5

let
    # Define the number of sites in the chain
    # The random MPS must be defined in the same Hilbert space as the Hamiltonian
    number_of_sites = 100
    sites = siteinds("S=1/2", number_of_sites)
    Psi0 = randomMPS(sites; linkdims = 2)

    # Define the paramters used in the Hamiltonian
    J = 1.0
    Delta = 0.5

    # Set up the Hamiltonian
    os = OpSum()

    # Set up the nearest-neighbor interactions
    for index = 1 : number_of_sites - 1
        os += 0.5 * J, "S+", index, "S-", index + 1
        os += 0.5 * J, "S-", index, "S+", index + 1
        os += Delta, "Sz", index, "Sz", index + 1
```

```
        end
        Hamiltonian = MPO(os, sites)

        # Parameters that control the DMRG optimization process
        nsweeps = 20
        maxdim = [20, 50, 200, 1000]
        cutoff = [1E-10]
        # Perform the DMRG calculation
        E0, Psi = dmrg(Hamiltonian, Psi0; nsweeps, maxdim, cutoff)

        println("The converged ground state energy is: $E0")
        println("")

        # Measure one-point and two-point functions
        Sx = expect(Psi, "Sx"; sites = 1 : number_of_sites)
        Cxx = correlation_matrix(Psi, "Sx", "Sx"; sites = 1 : number_of_sites)
        @show Sx


        # Store the wavefunction and expectation value of observables into a HDF5 file
        h5open("Data/XXZ_Delta$(Delta)_N$(number_of_sites).h5", "w") do file
            write(file, "Psi", Psi)
            write(file, "E0", E0)
            write(file, "Sx", Sx)
            write(file, "Cxx", Cxx)
        end

        return
    end
```

In this lecture note, we simulate the XXZ model on a 1D chain with OBC. We utilize a long chain and concentrate on the bulk region to extrapolate physical information, effectively removing edge effects. The convergence of a DMRG simulation is influenced by several factors, including whether the system is gapped or gapless, its proximity to a critical point, and its dimensionality. One way to check the convergence is to compute the variance, defined as $\langle\psi|\hat{H}^2|\psi\rangle - \Big(\langle\psi|\hat{H}|\psi\rangle\Big)^2 = \langle H^2\rangle - \langle H\rangle^2$, of the converged MPS to check whether it is an eigenstate by adding the following lines of codes.

```
# Compute the variance of the converged MPS
H2 = inner(Hamiltonian, Psi, Hamiltonian, Psi)
E = inner(Psi', Hamiltonian, Psi)
variance = H2 - E^2
@show variance
```

Entanglement entropy is one of the most fundamental quantities in quantum mechanics. For a bipartite quantum system with the Hilbert space $\mathcal{H} = \mathcal{H}_A \bigotimes \mathcal{H}_B$, we can quantify the entanglement between two subsystems A and B by the von Neumann entanglement entropy for a given quantum state $|\psi\rangle$,

$$S_{A|B}(|\psi\rangle) = -tr_A \hat{\rho}_A \ln \hat{\rho}_A = -tr_B \hat{\rho}_B \ln \hat{\rho}_B, \tag{10}$$

where $\hat{\rho}_A$ denotes the reduced density matrix of subsystem A and $\hat{\rho}_B$ denotes the reduced density matrix of subsystem B.

On the other hand, we can decompose a quantum state $|\psi\rangle$ using the *Schmidt decomposition*. For example, we can

bipartite a 1D chain into subsystems A and B and rewrite the quantum state $|\psi\rangle$ in a general form

$$|\psi\rangle = \sum_{i=1}^{\dim\mathcal{H}_A} \sum_{j=1}^{\dim\mathcal{H}_B} \psi_{ij} |i\rangle_A |j\rangle_B. \tag{11}$$

where the $\{|i\rangle_A\}$ and $\{|j\rangle_B\}$ form orthonormal bases of subsystems A and B respectively. As a result, we can compute the von Neumann entanglement entropy using

$$S_{A|B}(|\psi\rangle) = S_{vN} = -\sum_{\alpha=1}^{r} s_\alpha^2 \ln s_\alpha^2, \tag{12}$$

where s_α are the non-vanishing singular values if we perform an SVD of ψ. Using the MPS form, the von Neumann entanglement entropy can be computed naturally using the following block of codes.

```
# Compute the von Neumman entanglement entropy S_vN for a bipartition of the chain into two subsystems.
# Subsystem A consists of sites 1,2,...,b
# Subsystem B consists of sites b+1,b+2,...,N
function entanglement_entropy(input_psi :: MPS, site_index :: Int)
    SvN = 0.0
    s = siteinds(input_psi)
    orthogonalize!(input_psi, site_index)
    i, j = s[site_index], linkinds(input_psi, site_index - 1)
    _, S, _ = svd(input_psi[site_index], (j, i))
    for index in 1 : size(S, 1)
        p = S[index, index]^2
        SvN -= p * log(p)
    end
    return SvN
end
```

- **Exercise**: Set up an one-dimensional XXZ chain and apply open boundary condition. The length of the chain is set to $N = 100$. Explore the difference between the quasi-long-range order and long-range Ising-like antiferromagnetic order through spin correlation functions, setting $\Delta/J = 0.5$ and $\Delta/J = 1.5$ or any other valid values. Optional for advanced users: plot the entanglement entropy S_{vN} for each possible cut of the MPS. What do you observe?

- **Exercise:** Fig. 3 shows the spin correlation function $C_{xx}(\ell', \ell) = \langle S^x(\ell') S^x(\ell)\rangle$. Let's also consider other spin correlation functions $C_{yy}(\ell', \ell)$ and $C_{zz}(\ell', \ell)$. What do you observe in correlation function if there is a true long-range order?

C. The $J_1 - J_2$ Heisenberg Model

In this exercise, we use DMRG to obtain the ground state of the spin-1/2 $J_1 - J_2$ Heisenberg model in 1D. The Hamiltonian of the model is given by

$$H = J_1 \sum_{\langle i,j\rangle} \vec{S}_i \cdot \vec{S}_j + J_2 \sum_{\langle\langle i,j\rangle\rangle} \vec{S}_i \cdot \vec{S}_j, \tag{13}$$

where the sum is over all possible nearest-neighbor and second-nearest-neighbor pairs. Here $\vec{S}_i = (S_i^x, S_i^y, S_i^z)$ denotes 3-vectorial spin operators localized at the lattice site. In the 1D Heisenberg model ($J_2 = 0$), there can be no magnetic order due to the Hohenberg-Mermin-Wagner theorem [23–25]. The spin correlation function $C(r) = \langle \vec{S}_i \cdot \vec{S}_{i+r}\rangle$ decays with the distance r as $(-1)^r/r$ with a logarithmic correction. Adding the second-nearest-neighbor interaction introduce a quantum phase transition into a valence-bond solid state (VBS) when the interaction strength is strong enough. The spin correlation decays exponentially in the VBS state.

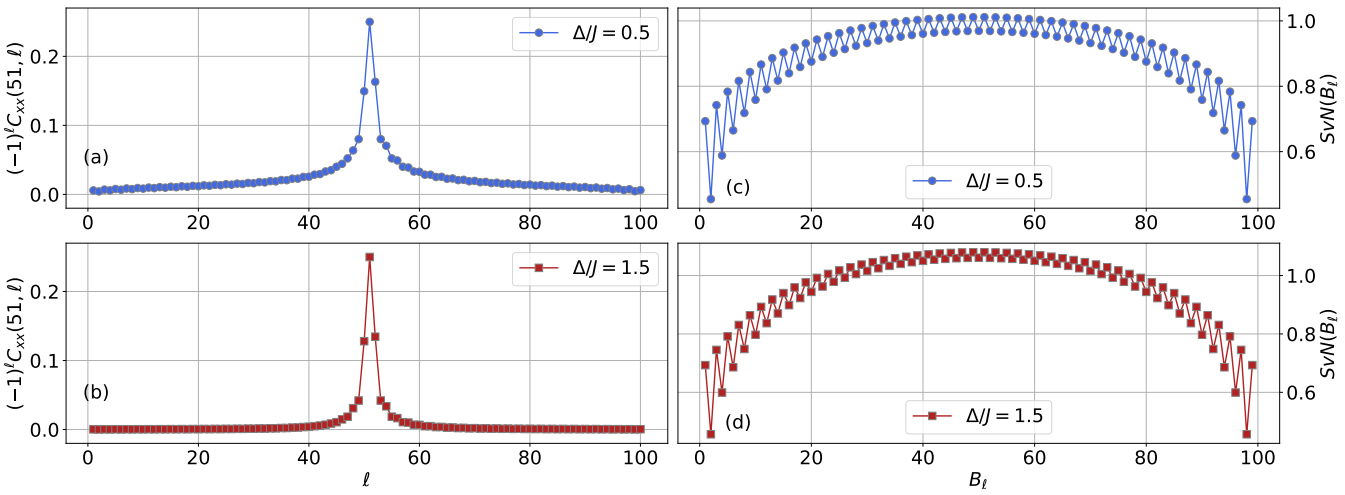


FIG. 3: Staggered spin correlation function $(-1)^\ell C_{xx}(\ell', \ell)$ for the XXZ model with (a) $\Delta/J = 0.5$ and (b) $\Delta/J = 1.5$. Using $\ell' = 51$ as the reference point for a chain with $N = 100$ sites. The von Neumann entanglement entropy $S_{vN}(B_\ell)$ along every possible cut B_ℓ in the MPS at (c) $\Delta/J = 0.5$ and (d) $\Delta/J = 1.5$.

- **Exercise:** Set up an one-dimensional Heisenberg chain with only nearest-neighbor interaction ($J_1 = 1, J_2 = 0$) and apply periodic boundary condition. The length of the chain is set to $N = 22$. Conduct a DMRG calculation to optimize the MPS and compute the ground-state energy. Compare the simulation result to the exact ground-state energy $E_0 = -9.78688065177$, which was obtained using brute-force exact diagonalization (ED).

- **Hint:** If you your setup is correct and you use a truncation error of $\epsilon = 10^{-10}$ and $\mathrm{N_{iter}} = 20$ iterations, the following is an example of the energy convergence history during a DMRG simulation. Upon convergence, the ground-state energy calculated in the DMRG simulation should closely approximate the value obtained through ED.

```
# Compute the variance of the converged MPS
After sweep 15 energy=-9.786880646351406  maxlinkdim=215 maxerr=9.85E-11 time=0.960
After sweep 16 energy=-9.786880646351435  maxlinkdim=215 maxerr=9.85E-11 time=0.812
After sweep 17 energy=-9.786880646351458  maxlinkdim=215 maxerr=9.85E-11 time=0.914
After sweep 18 energy=-9.786880646351449  maxlinkdim=215 maxerr=9.85E-11 time=0.832
After sweep 19 energy=-9.786880646351495  maxlinkdim=215 maxerr=9.85E-11 time=0.819
After sweep 20 energy=-9.78688064635151  maxlinkdim=215 maxerr=9.85E-11 time=0.858
```

- **Exercise:** As shown above, the ground-state energy computed using DMRG has already converged by iteration sweep 15. In such cases, we can terminate the DMRG calculation early based on specific criteria, such as the energy difference between two consecutive steps being smaller than a cutoff $\Delta E = |E_p - E_{p+1}| < \eta$, where p represents the number of iterations and $\eta = 10^{-9}$ is the cutoff set by you. To facilitate an early stop in a DMRG simulation, one needs to implement a customer observer and call DMRG with the customer observer. For more details, please refer to the document Early Stop.

- **Exercise**: By setting $J_2 = \frac{1}{2}J_1$, we obtain the Majumdar-Ghosh model that can be solved exactly,

$$H = J_1 \sum_{\langle i,j \rangle} \vec{S}_i \cdot \vec{S}_j + \frac{J_1}{2} \sum_{\langle\langle i,j \rangle\rangle} \vec{S}_i \cdot \vec{S}_j. \tag{14}$$

By completing the square, the Hamiltonian can be rewritten in an equivalent form

$$H = \frac{J_1}{4} \sum_{i=1}^{N} \left(\vec{S}_{i-1} + \vec{S}_i + \vec{S}_{i+1} \right)^2 + const. \tag{15}$$

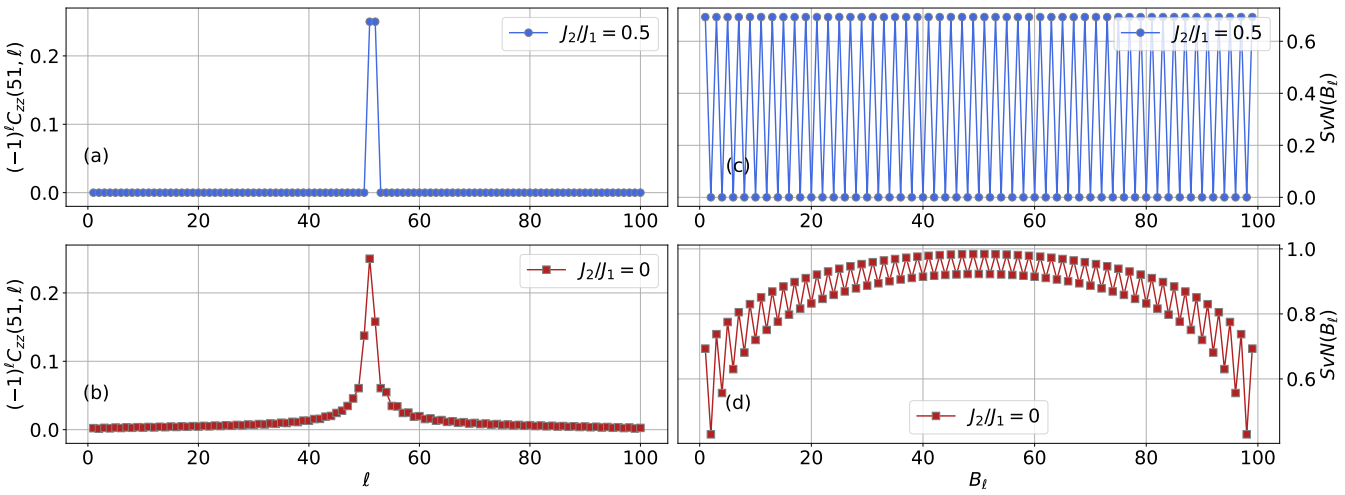


FIG. 4: Spin correlation function $(-1)^\ell C_{zz}(\ell', \ell)$ for the Heisenberg model with (a) $J_2/J_1 = 0.5$ and (b) $J_2/J_1 = 0$. Using $\ell' = 51$ as the reference point for a chain with $N = 100$ sites. The von Neumann entanglement entropy $S_{vN}(B_\ell)$ along every possible cut B_ℓ in the MPS at $J_2/J_1 = 0.5$ and $J_2/J_1 = 0$.

Conduct a DMRG calculation, using open boundary condition, to determine the energy per site in the ground state for the Majumdar-Ghosh model and compare the numerical value to the exact solution. Can you interpret the results using the concept of valence-bond solids? Additionally, observe the von Neumann entanglement entropy S_{vN} and the bond dimension χ. What insights can you derive from these observations? This article on Wikipedia might be useful Majumdar-Ghosh.

- **Exercise:** Although true symmetry-breaking phase transitions only occur in the thermodynamic limit (as the number of lattice sites $\to \infty$) for lattice models, the correlation functions and their Fourier transformations, known as structure factors, provide valuable insights into phase transitions on finite lattices. By employing careful finite-size scaling, phase transitions can be investigated using finite lattice sizes. Conduct DMRG simulations with OBC at $J_2/J_1 = 0$ and $J_2/J_1 = 0.5$ and calculate spin correlation functions $C_{xx}(\ell', \ell)$, $C_{yy}(\ell', \ell)$, $C_{zz}(\ell', \ell)$ as well as $C(\ell', \ell) = \sum_{\alpha=x,y,z} C_{\alpha\alpha}(\ell', \ell)$. Can you analyze and interpret the results?

- **Hint:** Fig. 4 shows the expected behavior for the staggered spin correlation function $(-1)^\ell C_{zz}(\ell', \ell)$ and the von Neumann entanglement entropy.

III. ACKNOWLEDGEMENT

B. X. would like to thank Miles Stoudenmire for insightful discussions and sharing his teaching materials. B. X. is supported by the U.S. Department of Energy, Office of Science, National Quantum Information Science Research Centers, Quantum Science Center.

[1] S. R. White, Density matrix formulation for quantum renormalization groups, Phys. Rev. Lett. **69**, 2863 (1992).
[2] S. R. White, Density-matrix algorithms for quantum renormalization groups, Phys. Rev. B **48**, 10345 (1993).
[3] S. R. White, Early days of dmrg, Nature Reviews Physics **5**, 264 (2023).
[4] S. Östlund and S. Rommer, Thermodynamic limit of density matrix renormalization, Phys. Rev. Lett. **75**, 3537 (1995).
[5] J. Dukelsky, M. A. Martín-Delgado, T. Nishino, and G. Sierra, Equivalence of the variational matrix product method and the density matrix renormalization group applied to spin chains, Europhysics Letters **43**, 457 (1998).
[6] U. Schollwöck, The density-matrix renormalization group in the age of matrix product states, Annals of Physics **326**, 96 (2011), january 2011 Special Issue.
[7] W. Hackbusch, *Tensor spaces and numerical tensor calculus*, Vol. 42 (Springer, 2012).
[8] U. Schollwöck, The density-matrix renormalization group, Rev. Mod. Phys. **77**, 259 (2005).
[9] U. Schollwock, 16 dmrg: Ground states, time evolution, and spectral functions, Emergent Phenomena in Correlated Matter (2013).

[10] R. Orús, Tensor networks for complex quantum systems, Nature Reviews Physics **1**, 538 (2019).
[11] F. Verstraete, T. Nishino, U. Schollwöck, M. C. Bañuls, G. K. Chan, and M. E. Stoudenmire, Density matrix renormalization group, 30 years on, Nature Reviews Physics **5**, 273 (2023).
[12] F. Verstraete and J. I. Cirac, Renormalization algorithms for quantum-many body systems in two and higher dimensions, arXiv preprint cond-mat/0407066 (2004).
[13] G. Vidal, Entanglement renormalization, Phys. Rev. Lett. **99**, 220405 (2007).
[14] G. Vidal, Efficient classical simulation of slightly entangled quantum computations, Phys. Rev. Lett. **91**, 147902 (2003).
[15] G. Vidal, Efficient simulation of one-dimensional quantum many-body systems, Phys. Rev. Lett. **93**, 040502 (2004).
[16] J. Haegeman, J. I. Cirac, T. J. Osborne, I. Pižorn, H. Verschelde, and F. Verstraete, Time-dependent variational principle for quantum lattices, Phys. Rev. Lett. **107**, 070601 (2011).
[17] J. Haegeman, C. Lubich, I. Oseledets, B. Vandereycken, and F. Verstraete, Unifying time evolution and optimization with matrix product states, Phys. Rev. B **94**, 165116 (2016).
[18] M. Fishman, S. R. White, and E. M. Stoudenmire, The ITensor Software Library for Tensor Network Calculations, SciPost Phys. Codebases , 4 (2022).
[19] M. Fishman, S. R. White, and E. M. Stoudenmire, Codebase release 0.3 for ITensor, SciPost Phys. Codebases , 4 (2022).
[20] I. P. McCulloch, Infinite size density matrix renormalization group, revisited, arXiv preprint arXiv:0804.2509 (2008).
[21] H. N. Phien, G. Vidal, and I. P. McCulloch, Infinite boundary conditions for matrix product state calculations, Phys. Rev. B **86**, 245107 (2012).
[22] V. Zauner-Stauber, L. Vanderstraeten, M. T. Fishman, F. Verstraete, and J. Haegeman, Variational optimization algorithms for uniform matrix product states, Phys. Rev. B **97**, 045145 (2018).
[23] P. C. Hohenberg, Existence of long-range order in one and two dimensions, Phys. Rev. **158**, 383 (1967).
[24] N. D. Mermin and H. Wagner, Absence of ferromagnetism or antiferromagnetism in one- or two-dimensional isotropic heisenberg models, Phys. Rev. Lett. **17**, 1133 (1966).
[25] N. D. Mermin and H. Wagner, Absence of ferromagnetism or antiferromagnetism in one- or two-dimensional isotropic heisenberg models, Phys. Rev. Lett. **17**, 1307 (1966).