Exercise 9: Density matrix renormalization group

This exercise uses the provided files `a_mps.py`,`b_model.py`,`tfi_exact.py` from last week and `d_dmrg.py`.

a) To use DMRG as discussed in the lecture, we need to write the Hamiltonian H as a matrix product operator (MPO). Extend the `TFIModel` class of `b_model.py` to generate a list of W tensors representing the MPO during the initialization. The model should save the MPO as a list `self.H_mpo`, containing one W entry for each site. Each W should be a numpy array with 4 indices in the order (bond index left, bond index right, local physical index ket, local physical index bra). At the boundaries, the MPO is automatically terminated in the DMRG code by

$$v_L = \begin{pmatrix} 1 & 0 & \dots \end{pmatrix} \text{ (left) and } \qquad v_R = \begin{pmatrix} \vdots \\ 0 \\ 1 \end{pmatrix} \text{ (right boundary).} \tag{1}$$

Hint: One can choose the W independent of the sites as

$$W = \begin{pmatrix} \mathbb{1} & \sigma^x & -g\sigma^z \\ 0 & 0 & -J\sigma^x \\ 0 & 0 & \mathbb{1} \end{pmatrix} \tag{2}$$

b) Read the code of `d_dmrg.py`. To run a DMRG simulation, you need to initialize an instance of the `DMRGEngine` class in `d_dmrg.py` and call it's `sweep` method a few times. Initialize an MPS with all spins pointing up, a model for $L = 14, J \equiv 1, g = 1.5$ and a `DMRGEngine`. How many sweeps does DMRG need to converge? Compare the energy of the resulting state after each sweep with the ground state energy obtained by exact diagonalization (`tfi_exact.py`).

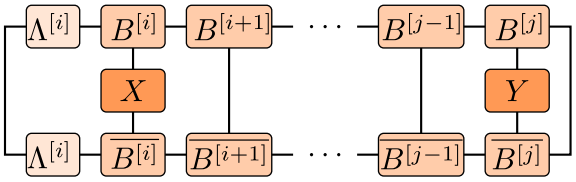
c) Write a function to perform a full DMRG simulation. The function should initialize an initial `MPS`, model and the `DMRGEngine` and perform sweeps until convergence. Finally, it should return the model and state.

d) Use this function to determine the half-chain entanglement entropy S (at the central bond) of the ground state for system sizes $L = 8, 16, 32, 64, 96, 128$ for $g = 1.5, 1.0, 0.5$ and $J = 1$ and plot it. At the critical point, the leading scaling with L is given by $S(L) = \frac{c}{6}\log(L)$, where c is the central charge. Extract the central charge c.

Hint: Make sure that you chose your bond dimension large enough to be converged with it. You can use `np.polyfit` to extract c if you fit a straight line to S vs. $\log(L)$. Use only a few points of the largest L available to get the dominant behaviour at large L.

e) Write a function to (efficiently) calculate (equal-time) correlation functions like $\langle\psi| X_i Y_j |\psi\rangle$ for some single-site operators X, Y, which are applied on sites i and j, respectively. The functions should take $|\psi\rangle, X, Y$ and i as an input and calculate the correlations $\langle\psi| X_i Y_j |\psi\rangle$ for all $j \geq i$.

Hint: Distinguish the cases $i = j$ and $i < j$. Assume that $|\psi\rangle$ is given in right canonical B form (since our DMRG implementation returns it as such). Use the orthonormality conditions of the canonical form on sites k for $k < i$ (left canonical form) and $k > j$ to simplify the expression of the correlation function to the following network (for $i < j$):

$\langle\psi| X_i Y_j |\psi\rangle =$ 

(3)

To keep track of the indices which you need to contract with `np.tensordot`, it can help to draw diagrams and label the legs. It is possible to write the function with a computational cost of $\mathcal{O}((L-i)^1)$, i.e., a single for loop of j.

f) Run DMRG for $g = 0.5, 1., 1.1, 1.2, 1.5$ and $L = 100$. Calculate and plot the correlations $\langle\sigma^x_{L/4}\sigma^x_j\rangle$. Does it agree with your expectations? How do the correlations decay at $g = 1$ and $g > 1$?

g) Plot the connected correlations $C(j) = \langle\sigma^x_{L/4}\sigma^x_j\rangle - \langle\sigma^x_{L/4}\rangle\langle\sigma^x_j\rangle$ versus $j - L/4$ on a logarithmic y-scale. For $g \neq 1$, extract the correlation length ξ by a fit $C(j) \propto \exp(-\frac{|j-L/4|}{\xi})$.

Hint: Again, you can use `np.polyfit` if you fit $\log(C(j))$ vs. $|j - L/4|$.