iDMRG

David Aceituno

Contents

iDMRG on C++

The code follows the steps outlined in the following paper

> Kjäll, J. A., Zaletel, M. P., Mong, R. S. K., Bardarson, J. H. & Pollmann, F. Phase diagram of the anisotropic spin-2 XXZ model: Infinite-system density matrix renormalization group study. *Phys. Rev. B* **87,** 235106 (2013).

and the file itebd.py.

Notation

Latin letters are reserved for physical indices, and greek indices for bond indices. The order of indices in all tensors is *physical indices* → *bond indices*. In diagrammatic notation we have

$$\Lambda_\alpha \beta = \alpha - \boxed{\Lambda} - \beta \;=\; 0 - \boxed{\Lambda} - 1 \tag{1}$$

$$\Gamma^i_{\alpha\beta} = \alpha - \overset{i}{\boxed{\Gamma}} - \beta \;=\; 1 - \overset{0}{\boxed{\Gamma}} - 2 \tag{2}$$

$$\theta^{ij}_{\alpha\beta} = \alpha - \overset{i \quad j}{\boxed{\quad\theta\quad}} - \beta \;=\; 2 - \overset{0 \quad 1}{\boxed{\quad\theta\quad}} - 3 \tag{3}$$

In C++ these are explicitly objects of type `Eigen::Tensor<double,rank,Eigen::ColMajor>;`, which are `typedef`'ed into shorthand `TensorR`, where `R` is the rank (0 to 4).

Model

Hamiltonian

We study the 1D Ising model with a transverse field, given by the following Hamiltonian

$$H = \frac{1}{2}\sum_i \left[-J\sigma_i^z\sigma_{i+1}^z - g\sigma_i^x\right].$$

or equivalently

$$H = \frac{1}{2}\sum_i \left[-J\sigma_i^z\sigma_{i+1}^z - \frac{g}{2}(\sigma_i^x + \sigma_{i+1}^x)\right].$$

For two sites this can be written out explicitly

$$H = \begin{pmatrix} J & -g/2 & -g/2 & 0 \\ -g/2 & -J & 0 & -g/2 \\ -g/2 & 0 & -J & -g/2 \\ 0 & -g/2 & -g/2 & J \end{pmatrix}.$$

and is obtained by the following code:

C++ code

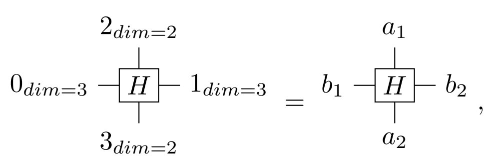
```
MatrixType H = 0.5*(2*J*kroneckerProduct(sz,sz) - g*(kroneckerProduct(sx,I) +
    kroneckerProduct(I,sx));
```

Hamiltonian in MPO form

In MPO form the Hamiltonian above reads

$$H_{\mathrm{MPO}} = \begin{pmatrix} I & 0 & 0 \\ \sigma^z & 0 & 0 \\ -g\sigma^x & -J\sigma^z & I \end{pmatrix}.$$

where each element is a 2×2 -matrix. To transform into an MPO note that this matrix is essentially an outer (3×3) matrix containing inner (2×2) matrices, or equivalently a $(2 \times 3) \times (2 \times 3)$ matrix. The goal is to obtain an MPO with dimensions $(3, 3, 2, 2)$, depicted as



(4)

where the left figure indicates the numbering of the indices and also their dimension.

Returning to H_{MPO}, remember that it should follow column-major notation in C++, such that the left (2×3) column is counted first. Calling these indices $(a_1 \times b_1)$, we note that a_1 should "tick" before b_1. Likewise for the (2×3) rows, denoted $(a_2 \times b_2)$, note that a_2 ticks before b_2. Therefore to get a rank 4 MPO we should simply reshape into indices $(a_1, b_1, a_2, b_2) = (2, 3, 2, 3)$.

Next we should reorder the indices to get the right diagram above. To do this, we do a transpose, or shuffle like $(1, 3, 0, 2)$ to get the order of indices $(b_1, b_2, a_1, a_2) = (3, 3, 2, 2)$. Then the first index b_1 selects the outer matrix row, and b_2 the outer matrix column, while a_1 selects inner row, and a_2 inner column.

A_α^j test