LAB XI
Simulating Adiabatic QC

Jakub Tworzydło

Institute of Theoretical Physics
Jakub.Tworzydlo@fuw.edu.pl

22/05/2023 Pasteura, Warszawa

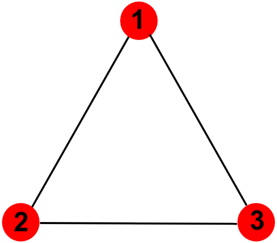
Task

Prepare the parameter dependent Hamiltonian matrix

$$H(\lambda) = (1 - \lambda)H_0 + \lambda H_1,$$

where

$$H_0 = -\sum_i S_i^x \text{ and } H_1 = -\sum_{ij} J_{ij} S_i^z S_j^z - \sum_i h_i S_i^z.$$



Take the above system with $h_1 = 0.6$, $h_2 = h_3 = 0$, $J_{12} = -1.1$, $J_{13} = -2.1$, $J_{23} = -3.8$ (all couplings are antiferromagnetic).

Note that $S_1^z = S^z \otimes \mathbf{1}_{2\times 2} \otimes \mathbf{1}_{2\times 2}$ etc. The tensor product $\otimes$ is just the Kronecker product of matrices, available in `Numpy`.

Task (continued)

Calculate and plot $\Delta E(\lambda) = E_1(\lambda) - E_0(\lambda)$, which is the difference between the first excited and the ground state energy.

Calculate the optimal running time of the adiabatic evolution T_{AQC}

$$T_{\mathrm{AQC}} = \int_0^1 \frac{d\lambda}{[\Delta E(\lambda)]^2}.$$

It is enough to approximate the integral by a discrete sum of small intervals.

Calculate and plot $\langle S_i^z \rangle$ for $i = 1, 2, 3$ as a function of λ, label the curves. Here $\langle S_i^z \rangle = \langle \psi | S_i^z | \psi \rangle$ is the expectation value of the operator S_i^z on the ground state eigenvector $|\psi(\lambda)\rangle$.

Hints

We need the following 2×2 spin matrices:

$$S^z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad S^x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

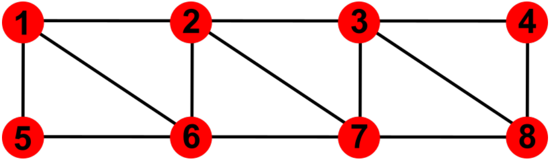
Spin operator S_i^z is a Kronecker product $S_i^z = \mathbf{1} \otimes \ldots \otimes S^z \otimes \ldots \otimes \mathbf{1}$ with 2×2 matrix S^z at position i. Implement $\otimes$ directly from `sicpy`:

```
from numpy import kron

sz  = np.array( [[1,0.],[0.,-1]] )
one = np.eye(2)

sz_1 = kron( sz, kron(one, one))
```

Extra Task

The purpose of this exercise it to write a more abstract code, which can store connectivity of the graph in a proper data structure. Introduce and use some data structure to encode the system graph. You can use e.g. a Python dictionary or tools from the package `networkx`.

Prepare the Hamiltonians H_0 and H_1 for the system:



Calculate and plot $\Delta E(\lambda)$ for the test case $J_{ij} = -1$ for all connections, and $h_i = 0.9$ for all magnetic fields.

Calculate $\Delta E(\lambda)$ for a few random realizations of J_{ij} and h_i and plot on a single picture. Draw the random values uniformly from the interval $[-1, 1]$.

Hints

Graph in Python can be represented e.g. by dictionary of lists

```
Graph = { 'Q1': ['Q2','Q4'], 'Q2': ['Q3'], ... }
Graph['Q1']
```

where a node key points to the nodes connected by edges.

One may also store the properties of a node in a dictionary

```
Node = { 'Q1': (sz1, sx1, h1), 'Q2': ... }.
```

Extra – open

There are many more things one can compute:

- entanglement entropy (as discussed in the **reading material**)
- qubism plots e.g. from `https://arxiv.org/abs/1112.3560`
- scaling the worst case gap with N for a given lattice (use `scipy.sparse` for sparse matrices).

If you are interested – one of the above could be your topic for the final presentation!