LAB XI
Simulating AQC in real time

Jakub Tworzydło

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

21/05/2024 Pasteura, Warszawa

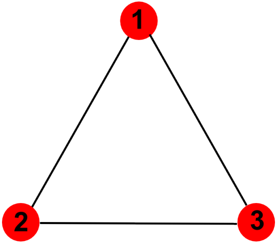
Task – intro

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 $h_1 = -0.5$, $h_2 = 0.5$,$h_3 = -0.1$, $J_{12} = -0.4$, $J_{13} = -1.6$, $J_{23} = -1.0$ (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 I

Calculate the ground state energy $E_{\mathrm{GS}}(\lambda)$ and the ground state eigenvector $|\psi(\lambda)\rangle$ as a function of λ.

Plot the expectation value $\langle S_i^z \rangle$ in the ground state for all spins $i = 1, 2, 3$. Label the curves on the plot, read the (optimal) result for $\lambda = 1$ from the graph.
Here $\langle S_i^z \rangle = \langle \psi(\lambda) | S_i^z | \psi(\lambda) \rangle$ is the expectation value of the operator S_i^z.

You may also plot E_{GS} to check as a benchmark.

Task II

Initiate the most democratic eigenstate ψ_+ (which is also the ground state at $\lambda = 0$); and initiate the final solution state ψ_f.

Using the eigenvalue decomposition of $H(\lambda_k)$ with $\lambda_k = t_k/T_m$ at every step, calculate the evolution of the initial state

$$\psi(t_k) = \exp\left(-iH(t_{k-1}/T_m)\ dt\right) \ldots \exp\left(-iH(t_0/T_m)\ dt\right)\psi_+$$

where $dt = T_m/M$ and $t_k = k\,dt$ for M disctere time steps.

Plot the expectation value $\langle S_i^z \rangle$ in the state $\psi(t_k)$ for all spins $i = 1, 2, 3$ as a function of time.

Plot (preferably panel plot) the success probability to find the desired optimum $|\langle \psi_f | \psi(t_k) \rangle|^2$ as a function of time. What is approximately the running time of the algorithm T_m to reach more then 90%?

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 `numpy`:

```
from numpy import kron
sz  = np.array( [[1,0.],[0.,-1]] )
one = np.eye(2)
sz_1 = kron( sz, kron(one, one)) # spin operator

v = np.array([1,1])/sqrt(2)
psiplus = kron( v, kron(v, v))   # democratic state
```

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 (or time T_{max}) with the number of spins on a ring (use `scipy.sparse` for sparse matrices)
- careful calculation of the Landau-Zener transition

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