Numerical Solution of Frustrated Quantum Heisenberg Spin Chain

Khagendra Adhikari[1*]
[1]University of Mississippi, Department of Physics and Astronomy, University, 38677

PHYS 730 FINAL PROJECT
Spring 2016

Introduction

The quantum Heisenberg spin chain is a simple one-dimensional model of magnetism that captures various complex physical phenomena. In this model, mutually interacting quantum spins are arranged with a regular spacing along a line at fixed positions.

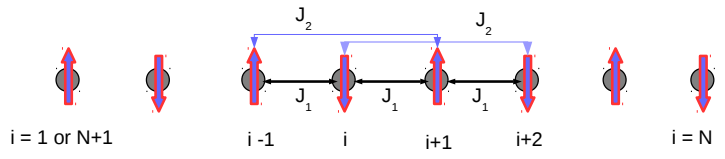

Fig. 1: Schematic representation of a frustrated quantum Heisenberg spin chain of length N.

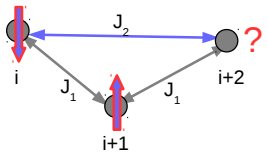

Fig. 2: The $(i+2)^{th}$ spin is frustrated. i.e. $\uparrow$ or $\downarrow$.

In Fig. 1, the sphere and the arrow represent the spin-half particle at the site i and its orientation, respectively, for finite system size $N = 8$. The strength of the exchange interaction J is isotropic, i.e. $\overleftrightarrow{J} = (J_x, J_y, J_z) = J(1, 1, 1)$. The nature of the interaction is *antiferromagnetic* $(J > 0)$, meaning that the minimum energy configuration of the system favors the opposite alignment of the spins. The simple and natural way to incorporate frustration is to consider the next-nearest-neighbor interation in the Hamiltoian. The nearest- and next-nearest-neighbor interactions favor incompatible spin orderings, and the competition between them gives rise to *frustration* as shown in Fig. 2.

*kadhikar@go.olemiss.edu

The Hamiltonian is

$$\hat{H} = \sum_{i=1}^{N} \left[J_1 \hat{\mathbf{S}}_i \cdot \hat{\mathbf{S}}_{i+1} + J_2 \hat{\mathbf{S}}_i \cdot \hat{\mathbf{S}}_{i+2} \right]. \tag{1}$$

$$\begin{aligned} \text{Here} \quad \hat{\mathbf{S}}_i \cdot \hat{\mathbf{S}}_{i'} &= \hat{S}_i^x \hat{S}_{i'}^x + \hat{S}_i^y \hat{S}_{i'}^y + \hat{S}_i^z \hat{S}_{i'}^z \\ &= \frac{1}{2} \left(\hat{S}_i^+ \hat{S}_{i'}^- + \hat{S}_i^- \hat{S}_{i'}^+ \right) + \hat{S}_i^z \hat{S}_{i'}^z, \end{aligned} \tag{2}$$

where $\hat{S}^{\pm} = \hat{S}^x \pm j\hat{S}^y$ are ladder operators. For spin S, ladder and cartesian operators are square matrices of dimension $2S+1$. They can be represented in the basis of the form $|S, m\rangle$ with $m = -S, \ldots, S$:

$$\hat{S}^{\pm}|S,m\rangle = \sqrt{S(S+1) - m(m \pm 1)}\,\hbar\,|S, m \pm 1\rangle, \qquad \hat{S}^z|S,m\rangle = m\hbar|S,m\rangle. \tag{3}$$

Non-Frustrated Model $(J_2/J_1 = 0)$:

In 1931, the model was solved analytically by Bethe [1] for any finite system size. Using Bethe's result, the exact ground state energy per site, $E_0/N = (1/4 - \log_e(2)) = -0.443147180559945$ in the limit of infinitely long chain was evaluated by Hulthen [2] in 1938. The ground state is magnetically disordered [3].

Frustrated Model $(J_2/J_1 = 0.5)$: **Majumdar Ghosh point (MGP)**

This is a special, exactly solvable point. The ground state has total spin $S_T = 0$ and energy per spin is $E_0/N = -3/8 = -0.375$ [5]. The ground state is doubly degenerate and has a simple structure.

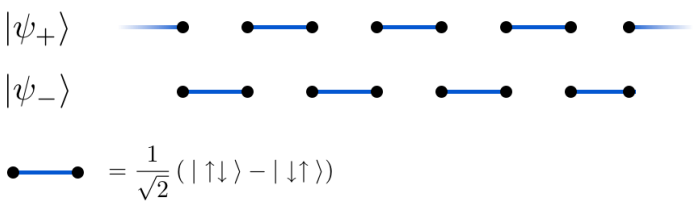

Fig. 3: The dumbbells represent singlets, the product of these singlets on alternating bond is an exact ground state.

The translational (discrete) symmetry is broken in the ground state.

$$|\psi_{k=0}\rangle = \frac{1}{\sqrt{2}} \left(|\psi_+\rangle + |\psi_-\rangle \right), \qquad |\psi_{k=\pi}\rangle = \frac{1}{\sqrt{2}} \left(|\psi_+\rangle - |\psi_-\rangle \right).$$

Numerical Method

The goal of this project is to solve the isotropic frustrated quantum Heisenberg spin-half chain. Solving the model means finding all of the eigenstates of the Hamiltonian and its corresponding eigenvalues. Spin-half particle can have spin up or down. So, any particle in the linear superposition $\alpha|\uparrow\rangle + \beta|\downarrow\rangle$ requires a two-dimensional *Hilbert space*. Therefore, the size of the Hamiltonian for N spin-half particles system is $2^N \times 2^N$.

The Hamiltonian matrix is generated (see Appendix A) and diagonalized for a finite-size system using Python programming language. The eigenvalues and eigenvectors represent the energy spectrum and coefficients of the basis vectors in the wave function respectively. Exact diagonalization studies are limited to rather small lattices because of the exponential increase of the basis size $n = (2S+1)^N$ with the number of spins, N. Hamiltonian matrix is a sparse matrix as shown in Fig. 4 and Fig. 5, a matrix in which most of the elements are zero. So, *scipy.sparse* A from *SciPy* is used which require less memory space to represent a sparse matrix, and allows for speeding up calculations involving these matrices.

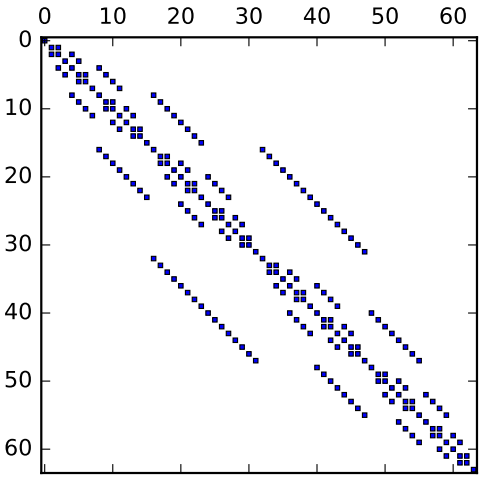

Fig. 4: Matrix showing non-zero elements in non-frustrated model for $N = 6$.

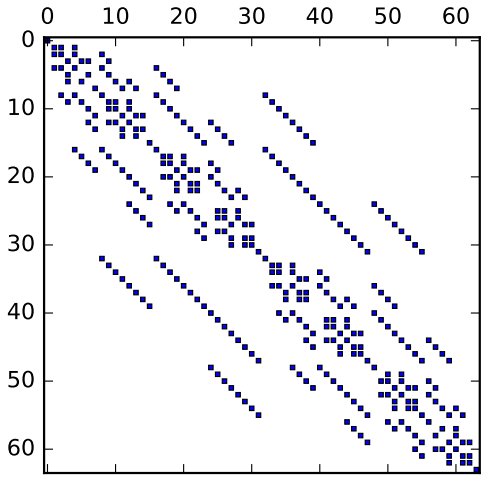

Fig. 5: Matrix showing non-zero elements in frustrated model for $N = 6$.

The simulation is carried out over a range of tuning parameter J_2/J_1, the ratio between the strength of next-nearest- and nearest-neighbor interactions, on finite-size systems of increasing size with result extrapolated to infinite-size limit. For this, the ground state energy per site versus square of the inverse chain length is graphed as shown in Fig. 7. When $N \to \infty$, $1/N^2 \to 0$, so the y-intercept is the ground state energy per site for the infinite-size system.

Result

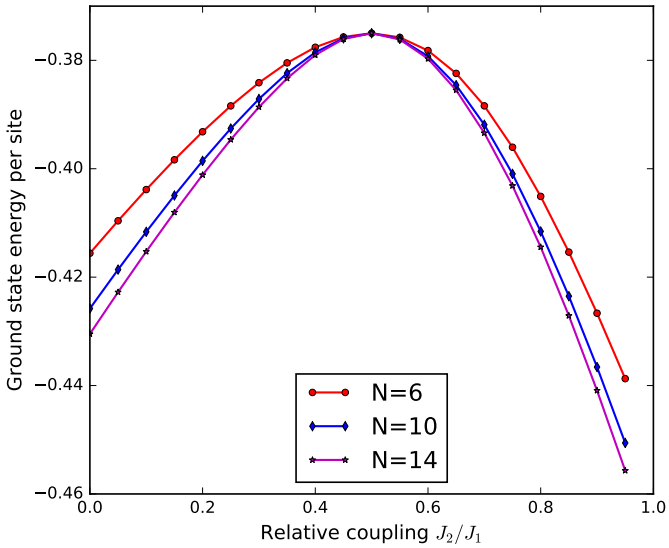

Fig. 6: The ground state energy per site versus relative coupling is plotted. Here, it has a finite size effect except at MGP ($J_2/J_1 = 0.5$) where ground state energy per site is $E_0/N = -0.375$, independent of system-size. For non-frustrated model ($J_2/J_1 = 0$), ground state energy per site is converging towards the Hulthen [2] result as system-size N increases.

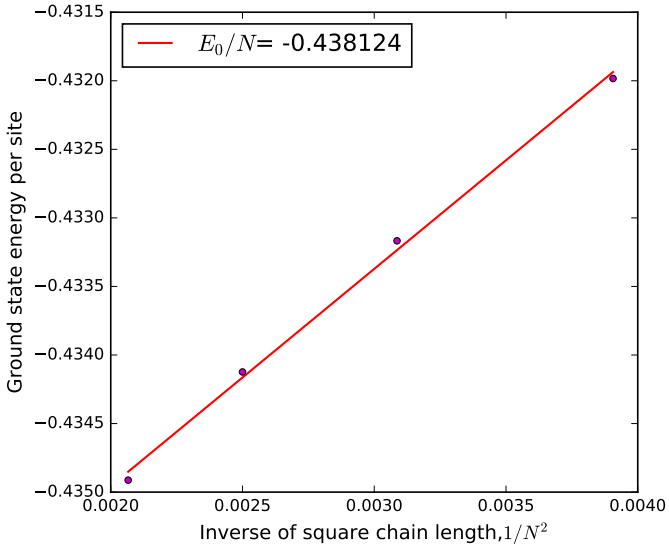

Fig. 7: For non-frustrated model ($J_2/J_1 = 0$), we can see that the ground state energy per site is linear with square of inverse chain length [4]. Extrapolating from finite clusters to infinite-size system, ground state energy per site $E_0/N = -0.438124$ is in close agreement with Hulten's known result [2] $E_0/N = -0.443147180559945$.

Model	Known results	My results
Non-frustrated model ($J_2/J_1 = 0$)	-0.443147180559945 [2]	-0.438124
Frustrated model at MGP ($J_2/J_1 = 0.5$)	-0.375 [5]	-0.375

Table 1: Result table for the ground state energy per site

Conclusion

The analytical solutions available for the non-frustrated model and the frustrated model at MGP give the testing ground to validate the algorithm. For non-frustrated model, the ground state energy per site extrapolated to the infinite-size system (thermodynamic limit) is in close agreement with the known result (see Table 1). In frustrated model, ground state energy per site at MGP is exactly equal to the Majumdar and Ghosh's result (see Table 1). This proves the validity of our results for the ground state energy per site over a range of relative coupling J_2/J_1. It can also be used for other *non-integrable* systems with only a slight modification. The models are not integrable for higher value of S. Therefore $S = 1/2$ is a special case for which analytical and numerical results can be compared. Using sparse matrix, the computation is carried out for bigger system size, $N = 22$, a matrix of size $4,194,304 \times 4,194,304$. Therefore extrapolated result is far better than that of storing a full matrix.

References

[1] H.A Bethe, *Z. Phys.* **71**, 205 (1931).

[2] L.Hulthen, *Ark. Mater. Astron. Fys. A* **26**, 1 (1938).

[3] D. Mermin and H. Wagner, *Phys. Rev. Lett.* **17** , 1133 (1966).

[4] Bonner, J.C., Fisher, M.E., *Phys. Rev.* **135**, A640 - A658 (1964).

[5] C. K. Majumdar and D. K. Ghosh, *J. Math. Phys.* **10**, 1388 (1969); ibid., 1399 (1969).

A Appendix

Snippet of code that generates a Hamiltonian matrix

```
def Hamiltonian(N,g):
    '''
        Inputs:    N = Numer of sites (Spins)
                   g = tuning parameter J_2/J_1
        Method:    Kronecker product
        Output:    H = Hamiltonian matrix that stores only non-zero matrix
                       elements in Compressed Sparse Row matrix format.
    '''
    H2,H2n=spin(g)
    SpId = ssp.csr_matrix(np.identity(MS))
    Sz = MS**N
    H = ssp.csr_matrix((Sz,Sz))
    temp1 = dict()
    temp2 = dict()
    for i in range(N-1):
        for j in range(N-1):
            if i == j:
                temp1[(i,j)]=H2
            else:
                temp1[(i,j)]=SpId
    Ha = ssp.csr_matrix((8,8))
    for i in range(N-2):
        for j in range(N-2):
            if i == j:
                temp2[(i,j)]=H2n
            else:
                temp2[(i,j)]=SpId
    Hb = ssp.csr_matrix((16,16))
    try:
        for i in range(N-1):
            for j in range(N-2):
                if j < 1:
                    Ha = ssp.kron(temp1[(i,j)],temp1[(i,j+1)],format='csr')
                else:
                    Ha = ssp.kron(Ha,temp1[(i,j+1)],format='csr')
            H = H + Ha
        if N > 3:
            for i in range(N-2):
                for j in range(N-3):
                    if j < 1:
                        Hb = ssp.kron(temp2[(i,j)],temp2[(i,j+1)],format='csr')
                    else:
                        Hb = ssp.kron(Hb,temp2[(i,j+1)],format='csr')
                H = H + Hb
        else:
            Hb = ssp.csr_matrix(temp2[(0,0)])
            H = H + Hb
    except MemoryError:
        print('The matrix you tried to build requires too much memory space.')
        return
    return H
```