Assignment 7

Quantum Information and Computing Course 2022/2023

Massimo Colombo

Department of Physics and Astronomy "Galileo Galilei"
DEGREE IN PHYSICS

December 20, 2022

ISING MODEL: THEORY

The quantum 1D-Ising model with **N ½-spin** particles and **transversal magnetic** field, has the following Hamiltonian:

$$\hat{H} = \lambda \sum_{i}^{N} \sigma_i^z + \sum_{i}^{N-1} \sigma_i^x \sigma_{i+1}^x$$

Where λ measures how strong the interaction with the external magnetic field is and **σ**_i corresponds respectively to the I-th pauli matrix:

$$\sigma_{\mathrm{x}} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad \sigma_{\mathrm{z}} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

In order to solve the problem numerically (finding the eigenvalues), we first have to construct the 2^N x 2^N matrix associated with the Hamiltonian. To do this, **Kronecker product** must be used:

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & \cdots & a_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathbf{B} & \cdots & a_{mn}\mathbf{B} \end{bmatrix}$$

In this way, the operators in the Hamiltonian can be represented as follows:

$$\sigma_z^i = \mathbb{I}_2^1 \otimes \mathbb{I}_2^2 \otimes \ldots \otimes \mathbb{I}_2^{i-1} \otimes \sigma_z \otimes \mathbb{I}_2^{i+1} \otimes \ldots \otimes \mathbb{I}_2^N$$

$$\sigma_x^i \sigma_x^{i+1} = \mathbb{I}_2^1 \otimes \mathbb{I}_2^2 \otimes \ldots \otimes \mathbb{I}_2^{i-1} \otimes \sigma_x^i \otimes \sigma_x^{i+1} \otimes \mathbb{I}_2^{i+2} \otimes \ldots \otimes \mathbb{I}_2^N$$

0em

CODE STEPS: 1

CODE STEP: **Kronecker product** ⟶ **N-Dim Identity** ⟶ **Ising Hamiltonian** ⟶ **Diagonalization**

1) KRONECKER PRODUCT:

```
interface operator(.tens.)
   module procedure matA_tens_matB
end interface

function matA_tens_matB(AA, BB) result(CC)
   complex*16, dimension(:,:), intent(IN)  :: AA, BB
   complex*16, allocatable, dimension(:,:) :: CC
   integer*4, dimension(2)                 :: dimAA, dimBB, dimCC
   integer*4                               :: iiA, jjA, iiB, jjB, iiC, jjC

   dimCC(:) = (/dimAA(1)*dimBB(1), dimAA(2)*dimBB(2)/)
   DO jjB = 0, dimBB(2) -1
      DO iiB = 0, dimBB(1) -1
         DO jjA = 1, dimAA(2)
            DO iiA = 1, dimAA(1)
            iiC = iiA + iiB * dimAA(1)
            jjC = jjA + jjB * dimAA(2)
            CC(iiC, jjC) = AA(iiA, jjA) * BB(iiB +1, jjB +1)
```

To be coherent with 6th assignment's representation of a MB-wavefunction, the kronecker product on the side returns the tensor product of 2 matrices with inverted order (A.tens.B ⟶ B.tens.A). This has no consequences due to the tensor product property*.

*In general, $\mathbf{A} \otimes \mathbf{B}$ and $\mathbf{B} \otimes \mathbf{A}$ are different matrices. However, $\mathbf{A} \otimes \mathbf{B}$ and $\mathbf{B} \otimes \mathbf{A}$ are permutation equivalent, meaning that there exist permutation matrices $\mathbf{P}$ and $\mathbf{Q}$ such that: $\mathbf{B} \otimes \mathbf{A} = \mathbf{P}\,(\mathbf{A} \otimes \mathbf{B})\,\mathbf{Q}$.

2) N-DIM IDENTITY:

```
function d_power_N_Id(local_dim, Nbodies) result(big_identity)
      integer*4                               :: local_dim, Nbodies, size, ii
      complex*16, allocatable, dimension(:,:) :: big_identity

      size = local_dim**Nbodies
      ALLOCATE(big_identity(size, size))
      big_identity = 0d0
      DO ii = 1, size
          big_identity(ii, ii) = 1d0
      END DO
```

The function returns an identity matrix with dimension D^N x D^N.
Given that the Kronecker product of Identity matrices is itself an Identity matrix (with higher dimension), this function will spare some tensor product calculation in the following.

0em

CODE STEPS: 2

3) ISING HAMILTONIAN:

```
function get_Ising_Hamiltonian(Nbodies, Lambda) result(Ising_ham)
    integer*4                                   :: Nbodies, size, NN
    complex*16, dimension(2, 2)                 :: pauliZ, pauliX
    complex*16, allocatable, dimension(:,:)     :: NI_ham, I_ham, Ising_ham
    real*8                                      :: Lambda

    pauliZ(1, :) = ((/1d0, 0d0/))
    pauliZ(2, :) = ((/0d0, -1d0/))
    pauliX(1, :) = ((/0d0, 1d0/))
    pauliX(2, :) = ((/1d0, 0d0/))

    size = 2**Nbodies
    ALLOCATE(NI_ham(size, size), I_ham(size, size), Ising_ham(size, size))
    DO NN = 1, Nbodies
       NI_ham = NI_ham + (d_power_N_Id(2, NN-1).tens.(pauliZ)).tens.d_power_N_Id(2, Nbodies - NN)
    END DO
    DO NN = 1, Nbodies -1
       I_ham = I_ham + ((d_power_N_Id(2, NN-1).tens.(pauliX)).tens.pauliX).tens.d_power_N_Id(2, Nbodies - NN -1)
    END DO
    Ising_ham = Lambda * NI_ham + I_ham
```

After allocating the right size matrices, the tensor product showed in the 1st slide is translated into the code.

Non interacting part, (Kronecker product involving z-pauli matrix).

Interacting part, (Kronecker product involving x-pauli matrices).

4) DIAGONALIZATION:

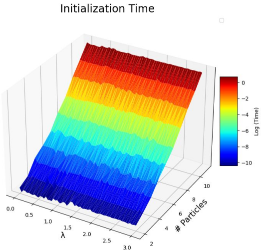
```
function diagonalize_herm_mat(hermitian_mat) result(eigenvalues)
    complex*16, dimension(:,:)              :: hermitian_mat
    complex*16, allocatable, dimension(:) ::  WORK, RWORK, pre_WORK
    real*8, allocatable, dimension(:)       :: eigenvalues
    integer :: INFO, LWORK, dim

    dim = size(hermitian_mat, 1)
    ALLOCATE(eigenvalues(dim), pre_WORK(1), RWORK(max(1, 3*dim-2)))
    LWORK = -1  !With LWORK = -1, it optimizes the dimension of WORK array. Returning pre_WORK(1).
    CALL zheev("N","U", dim, hermitian_mat, dim, eigenvalues, pre_WORK, LWORK, RWORK, INFO)
    LWORK = int(pre_WORK(1))    !After allocating WORK with optimal dimension it can compute the eigenvalues:
    ALLOCATE(WORK(MAX(1,LWORK)))

    CALL zheev("N","U", dim, hermitian_mat, dim, eigenvalues, WORK, LWORK, RWORK, INFO)
```

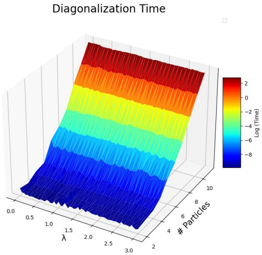
Lapack's routine Zheev has been utilized to diagonalize a given hermitian matrix.

0em

RESULTS: 1

Thanks to a python script, it was possible to run the program with different values of **λ** and number of particles.
To understand the code's behavior the function **CPU_time( )** has been called.
Ising hamiltonian **initialization** and **diagonalization** times are presented in the following:





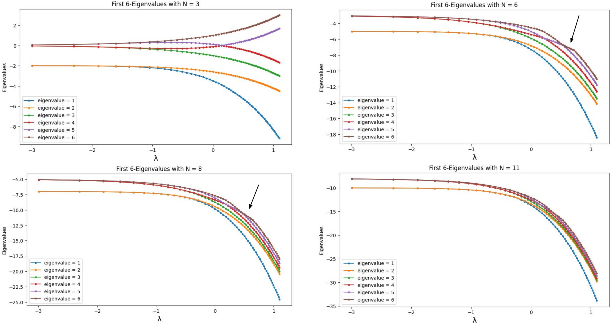
In the plots, we can observe that **diagonalizing** the hamiltonian is the part that **requires more time**. For 11 particles, it requires about 10 seconds to diagonalize it. The laptop, in which the script has been run, achieved to complete the execution with 12 particles. However, half of the times, it crashed. For that reason **Nmax** is considered to be **11**.
In the following, Ising hamiltonian with 2 particles and **λ** = 1:

```
Ising Hamiltonian:
   2.00000  +  0.00000i     0.00000  +  0.00000i     0.00000  +  0.00000i     1.00000  +  0.00000i
   0.00000  +  0.00000i     0.00000  +  0.00000i     1.00000  +  0.00000i     0.00000  +  0.00000i
   0.00000  +  0.00000i     1.00000  +  0.00000i     0.00000  +  0.00000i     0.00000  +  0.00000i
   1.00000  +  0.00000i     0.00000  +  0.00000i     0.00000  +  0.00000i    -2.00000  +  0.00000i
```

0em

RESULTS: 2

For different amounts of particles, the **first 6 eigenvalues** of the Ising hamiltonian have been plotted as function of **λ**.



RESULTS

For **λ** = 0, we observe degeneracy in the eigenvalues, as expected "turning off" the external field.
Increasing the magnetic field, the degeneracy disappears and the energy levels split.
We expect a phase transition for the one-dimensional solution with the transverse field for **λ** = 1.
In the plot with N = 6, 8, a non-smooth behavior around that value is more evident. The plots are considered correct considering the theoretical expectation.