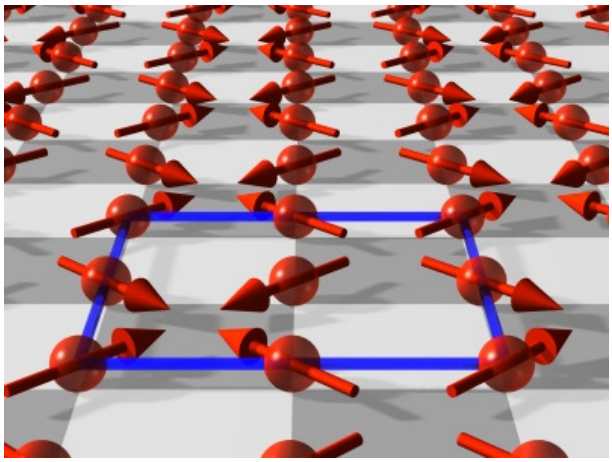
Master degree in Physics & Computational Physics

Quantum dynamics and quantum control
Lattice spin systems

Bruno Bellomo: based on previous material from David Viennot and Quentin Ansel



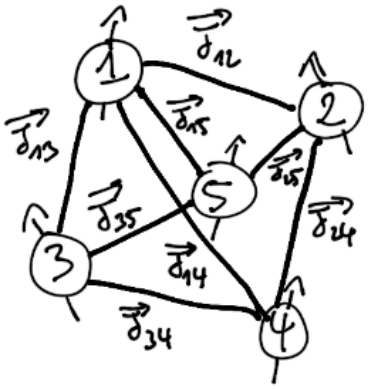
Contents

Chapter 1

Introduction

1.1 Goals

This course is dedicated to the study of the lattice spin systems, which can be defined as undirected graphs where the nodes (the vertices) are quantum spins and where the edges (the arcs) are decorated by vectors $\vec{J} = (J^x, J^y, J^z)$ called exchange integrals and modeling the strengths in the three directions of the spin-spin interactions (for the two spins linked by the edge). $\vec{J} = \vec{0}$ is equivalent to no edge between the two nodes.



Such quantum systems occur in different physical contexts:

- in dense matter physics where they model the magnetic properties of matter ($J > 0$ for para- and ferromagnetic materials, $J < 0$ for antiferromagnetic materials);
- in biophysics where they model NMR systems;
- in nanotechnologies for spintronics or magnonics devices, characterized by spin currents induced by the transport of electrons or of magnons (spin waves);
- in quantum field theory where some discrete space-time approximations (lattice gauge theories) take a lattice spin structure;
- in quantum information theory where $\frac{1}{2}$-spin systems are identified with qubits and where lattice spin systems are identified to qubit registers or to the whole of a quantum computer.

In the context of this course we restrict our attention only on $\frac{1}{2}$-spin systems.

Our main focus here is represented by the the physical properties of the lattice spin systems. In dense matter applications of lattice spin systems, interesting properties of the materials are encoded in the spectral properties of the quantum lattice Hamiltonian. In order to enlighten these properties, we also present in this course the numerical methods to compute and to analyse the spectra of hermitian matrices.

In this course, we do not take into account thermal effects onto the lattice spin systems. The question of thermal effects onto a quantum system is the subject of the part of the course on "open quantum systems".

In consequences, the models treated in this manuscript are experimentally valid at very small temperatures (in the neighborhood of $T = 0$ K). The question of quantum system cooling is viewed in the part of the course on "cold atoms". A presentation of general computational strategies of quantum control, not treated here, is instead realized in the part of the course on "optimal control".

1.2 The models of lattice spin systems

1.2.1 State space and Hamiltonian

The Hilbert space of a $\frac{1}{2}$-spin is $\mathbb{C}^2$ endowed with the canonical basis $(|s = 1/2, m_s = 1/2\rangle, |s = 1/2, m_s = -1/2\rangle)$ often written $(|\uparrow\rangle, |\downarrow\rangle)$. $|\uparrow\rangle$ is the state where the spin is parallel to the z-direction and $|\downarrow\rangle$ is the state where it is antiparallel. In this manuscript, in order to use quantum information theory, we use the binary notation $|0\rangle \equiv |\uparrow\rangle$ and $|1\rangle \equiv |\downarrow\rangle$ corresponding to the two fundamental states of a qubit. The spin observables are defined by the matrices (in the basis $\{|0\rangle, |1\rangle\}$):

$$S_x = \frac{1}{2}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad S_y = \frac{1}{2}\begin{pmatrix} 0 & -\mathbb{i} \\ \mathbb{i} & 0 \end{pmatrix}, \quad S_z = \frac{1}{2}\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{1.1}$$

Note that throughout this manuscript we use the atomic units defined by $\hbar = 1$ a.u.. The z-direction must be defined by the direction of an external magnetic field $\vec{B} = B\vec{e}_z$, and the Hamiltonian of an isolated spin is (by Zeeman effect)

$$h_0 = -\gamma B S_z = \begin{pmatrix} -\frac{\omega}{2} & 0 \\ 0 & \frac{\omega}{2} \end{pmatrix}, \tag{1.2}$$

where γ is the gyromagnetic ratio of the particle supporting the spin, and $\omega > 0$ is the Larmor precession frequency. Since the Hamiltonian is defined up to a constant, it is often interesting to consider

$$h_0 = \begin{pmatrix} -\omega & 0 \\ 0 & 0 \end{pmatrix}, \tag{1.3}$$

in order to have a negative spectrum. The state $|0\rangle$ is then the ground state associated to the above Hamiltonian, while $|1\rangle$ is the excited stae.

A system of N spins is characterized by the Hilbert space $\mathcal{H} = (\mathbb{C}^2)^{\otimes N} = \mathbb{C}^2 \otimes ... \otimes \mathbb{C}^2$, $\dim \mathcal{H} = 2^N$. The tensor basis vectors are defined by $|ab\rangle \equiv |a\rangle \otimes |b\rangle$ (with $a, b \in \{0, 1\}$). The canonical basis of $\mathcal{H}$ is chosen as being the tensor basis sorted with respect to the binary number sequence. For example, the canonical basis of $(\mathbb{C}^2)^{\otimes 2}$ is chosen to be $(|00\rangle, |01\rangle, |10\rangle, |11\rangle)$. It could be also usefull to denote an element of the basis by its position in the basis, it is the decimal notation $|d\rangle_{10} = |b\rangle$ where b and d are the same number written respectively in base-2 and in base-10. For example with $N = 2$, $|00\rangle = |0\rangle_{10}$, $|01\rangle = |1\rangle_{10}$, $|10\rangle = |2\rangle_{10}$, and $|11\rangle = |3\rangle_{10}$. Since no ambiguities are possible, we will omit the subscript '10' in the notation.

The interaction Hamiltonian between two spins is

$$h_{int} = -\sum_{u \in \{x,y,z\}} J^u S_u \otimes S_u \equiv -\vec{J} \cdot \vec{S} \odot \vec{S} \tag{1.4}$$

$\vec{J}$ is the exchange integral vector. The value of J^u is determined by the wave functions of the particles supporting the two spins and by the space distance between them. J^u decreases with the distance between the two particles.

For a system of $N = 3$ spins, the general Hamiltonian is then $H = H_0 + H_{int}$ where

$$H_0 = h_0 \otimes \mathrm{id}_2 \otimes \mathrm{id}_2 + \mathrm{id}_2 \otimes h_0 \otimes \mathrm{id}_2 + \mathrm{id}_2 \otimes \mathrm{id}_2 \otimes h_0, \tag{1.5}$$

id_2 denoting the 2×2 identity matrix, and

$$H_{int} = -\vec{J}_{12} \cdot \vec{S} \odot \vec{S} \otimes \mathrm{id}_2 - \vec{J}_{13} \cdot \vec{S} \otimes \mathrm{id}_2 \odot \vec{S} - \vec{J}_{23} \cdot \mathrm{id}_2 \otimes \vec{S} \odot \vec{S}. \tag{1.6}$$

In order to simplify the notation, we omit the identity matrices and we add a subscript in order to specify the position of the operator into the tensor sequence:

$$H_0 = h_{01} + h_{02} + h_{03}, \tag{1.7}$$

$$H_{int} = -\vec{J}_{12} \cdot \vec{S}_1 \odot \vec{S}_2 - \vec{J}_{13} \cdot \vec{S}_1 \odot \vec{S}_3 - \vec{J}_{23} \cdot \vec{S}_2 \odot \vec{S}_3. \tag{1.8}$$

For a system with N spins, we have then

$$H_0 = \sum_{i=1}^{N} h_{0i}, \tag{1.9}$$

$$H_{int} = -\sum_{i=1}^{N} \sum_{j>i} \vec{J}_{ij} \cdot \vec{S}_i \odot \vec{S}_j = -\sum_{i=1}^{N} \sum_{j>i} \sum_{u \in \{x,y,z\}} J_{ij}^u \cdot S_{ui} \otimes S_{uj}. \tag{1.10}$$

1.2.2 The different kinds of lattice spin systems

A lattice spin system is said

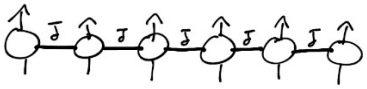
- ferromagnetic if $\forall i, j, u,\ J_{ij}^u \geq 0$;
- antiferromagnetic if $\forall i, j, u,\ J_{ij}^u \leq 0$;
- Ising-Z if $\forall i, j,\ J_{ij}^x = J_{ij}^y = 0$ and $J_{ij}^z = J$ (for a non-zero constant J);
- Ising-X if $\forall i, j,\ J_{ij}^y = J_{ij}^z = 0$ and $J_{ij}^x = J$ (for a non-zero constant J);
- Heisenberg-XXX if $\forall i, j, u,\ J_{ij}^u = J$ (for a constant J);
- Heisenberg-XXZ if $\forall i, j,\ J_{ij}^x = J_{ij}^y = J^x$ and $J_{ij}^z = J^z$ (for non-zero constants J^x and J^z);
- Heisenberg-XYZ if $\forall i, j,\ J_{ij}^x = J^x$, $J_{ij}^y = J^y$ and $J_{ij}^z = J^z$ (for non-zero constants J^x, J^y and J^z).

For Ising-Z spin systems we have $H_{int} = -J \sum_{i=1}^{N} \sum_{j>i} S_{zi} \otimes S_{zj}$ and for Heisenberg-XXX spin systems we have $H_{int} = -J \sum_{i=1}^{N} \sum_{j>i} \vec{S}_i \odot \vec{S}_j$.

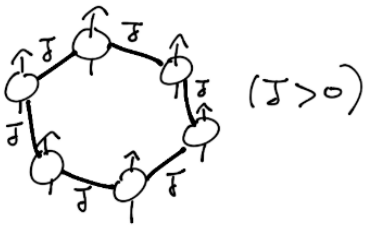
1.2.3 The different topologies of lattice spin systems

We can distinguish different topologies of lattice spin systems corresponding to different topologies of the graph:

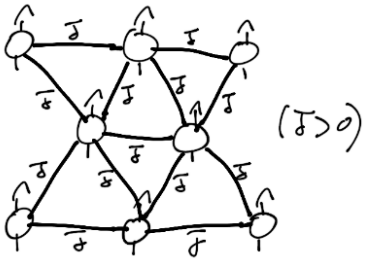
- open spin chain



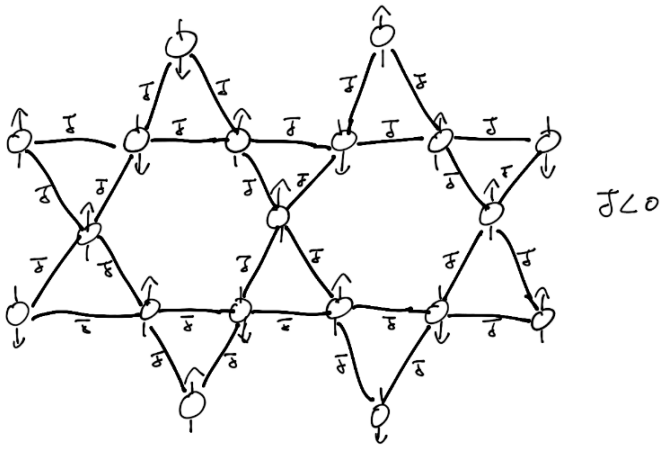
- closed spin chain



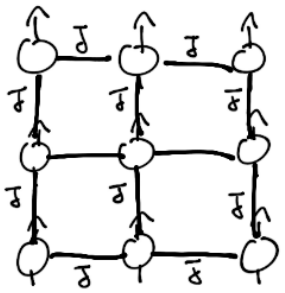
- triangular spin lattice (with ferromagnetic interaction)



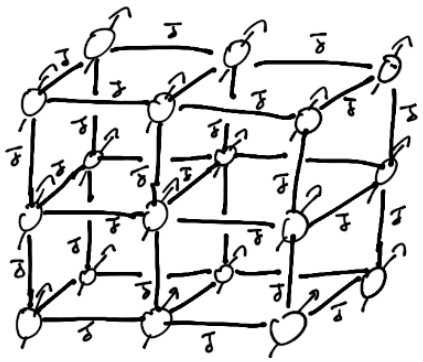
- spin liquid (antiferromagnetic triangular spin lattice as TaS_2 or antiferromagnetic kagome, i.e., trihexagonal tiling, spin lattice as $Rb_2Cu_3SnF_{12}$)



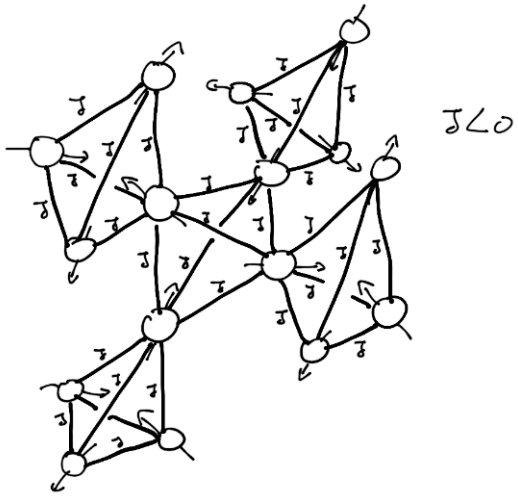
- rectangular spin lattice



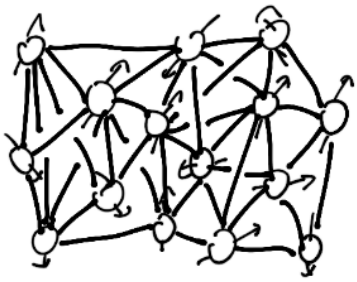
- cubic spin lattice



- spin ice (antiferromagnetic tetrahedron spin lattice as $Dy_2Ti_2O_7$ and $Ho_2Ti_2O_7$)



- spin glass (where J_{ij}^u are positive and negative random values) corresponding to disordered materials.



1.3 Numerical tools

1.3.1 General considerations

The numerical simulations will be coded with the `python` language. We need some libraries: `numpy` for numerical calculus, `linalg` for linear algebra functions, `matplotlib` for graphical representations and `random` for pseudo-random number generators.

```
import numpy as np;
from numpy import linalg as la;
import matplotlib.pyplot as py;
import random as rand;
```

Complete documentations concerning `numpy`, `matplotlib` and `random` are avaible at:
`https://docs.scipy.org/doc/`
`https://matplotlib.org`
`https://docs.python.org/2/library/random.html`

We need also to convert decimal numbers to binary numbers and reciprocally. Binary numbers are represented by strings. For example to convert the binary number 001001 to its decimal value 9:

```
int('001001',2)
```

```
9
```

and to convert 9 to its binary representation:

```
format(9,'06b')
```

```
'001001'
```

The `format` label `'0xb'` is build as follows: `x` is the number of digits of the representation, the first `0` indicates that the non-used digits appear as 0, and `b` specifies the binary representation. In the program, the number of digits is the number of spins and is loaded in a python variable `nspin`. In that case, the conversion takes

the form:

```
format(9,'0'+str(nspin)+'b')
```

It is important to distinguish the two different python commands list and array. A list is a collection of objects obeying to the Boolean algebra rules, whereas an array is a collection of numbers obeying to the linear algebra rules. The main difference is the following:

```
[1.,0.]+[0.,1.]
```

```
[1.,0.,0.,1.]
```

The operator + corresponds to the union (concatenation) for the lists.

```
np.array([1.,0.])+np.array([0.,1.])
```

```
array([1.,1.])
```

The operator + corresponds to the vector addition for the arrays. Moreover

```
2.*np.array([1.,0.])
```

```
array([2.,0.])
```

The operator * between a number and an array corresponds to the multiplication of a vector by a scalar. * is not defined for the lists. We can convert a list `l` to an array and an array `a` to a list as follows:

```
a=np.array(l); l=a.tolist()
```

All objects of the linear algebra are represented by arrays, the vectors (type $(1,0)$ or $(0,1)$) are the conversions of lists of numbers, the matrices [type $(1,1)$] are the conversions of lists of lists, the tensors [type $(2,1)$ for example] are the conversions of lists of lists of lists,... The 2×2 identity matrix can be then built as

```
id2 = np.array([[1.,0.],[0.,1.]]);
```

Note that a `numpy` function builds directly the identity matrix:

```
id2 = np.identity(2);
```

Finally, a very useful function is the one that return the tensor product of two matrices A, B in a matrix form (usually referred as the Kronecker product):

```
AkronB = LA2.kron(A,B);
```

1.3.2 The DQCspinlattice file

To study the spin lattices, we need functions to build quantum operators of the spin system and to compute the quantum information properties of the states. Several functions are already defined in the DQCspinlattice file which will be at disposal. The functional requirements of these functions are the followings:

- **Tensor product functions:**

  `def tensorvect(`a`,`b`)`
  input data: a, b: 1D arrays representing two vectors.
  output: 1D array representing $a \otimes b$ in the linear representation.

  `def tensorvectop(`a`,`b`)`
  input data: a, b: 2D square arrays representing two operators.
  output: 2D square array representing $a \otimes b$ in the linear representation.

  `def opchain(`a`,`i`,nspin)`
  input data: a: 2×2 array representing an operator.
  i: integer corresponding to the number of the spin on which a acts.
  `nspin`: number of spins of the lattice (integer).
  output: 2D square array representing $a_i = \mathrm{id}_{2^{i-1}} \otimes a \otimes \mathrm{id}_{2^{N-i}}$.

  `def opchain2(`a`,`i`,`b`,`j`,nspin)`
  input data: a: 2×2 array representing an operator; b: 2×2 array representing an operator.
  i and j: integers corresponding to which spins a and b, respectively, act.
  `nspin`: number of spins of the lattice (integer).
  output: 2D square array representing $a_i b_j = \mathrm{id}_{2^{i-1}} \otimes a \otimes \mathrm{id}_{2^{j-(i+1)}} \otimes b \otimes \mathrm{id}_{2^{N-j}}$.

- **State manipulation functions:**

  `def buildstate(bin)`

input data `bin`: binary number (as a string).
output 1D array representing the state $|bin\rangle$.

This function acts as follows:

```
buildstate('101')
array([0.,0.,0.,0.,0.,1.,0.,0.])
```

`def diracrep(psi,nspin)`
input data `psi`: 1D array representing a vector.
`nspin`: number of spins of the lattice (integer).
output string of the form $c_{000}|000\rangle + c_{001}|001\rangle + c_{010}|010\rangle + c_{011}|011\rangle + c_{100}|100\rangle + c_{101}|101\rangle + c_{110}|110\rangle + c_{111}|111\rangle$ (for $N = 3$ in this example), where $c_{ijk} = \langle ijk|\psi\rangle$ (the values with $|c_{ijk}| < 10^{-6}$ are ignored and do not appear in the string).

For example this function acts as follows:

```
diracrep(np.array([0.,0.,1.,0.,0.,1.,0.,0.])/np.sqrt(2.),3)
'+0.707106781187|010>+0.707106781187|101>'
```

`def densmat(psi,`i`,nspin)`
input data `psi`: 1D array representing the lattice state.
i: a spin number (integer).
`nspin`: number of spins of the lattice (integer)
output 2×2 array representing the density matrix of i-th spin ($\rho_i = \mathrm{tr}_{1,2,\ldots,\hat{i},\ldots,N}|\psi\rangle\langle\psi|$ where $\hat{i}$ means deprived of i).

`def avdensmat(psi,nspin)`
input data `psi`: 1D array representing the lattice state.
`nspin`: number of spins of the lattice (integer)
output 2×2 array representing the average density matrix $\frac{1}{N}\sum_{i=1}^{N}\rho_i$ where ρ_i is the density matrix of the i-th spin.

- **Quantum information theory functions:** `def purity(rho)`
  input data `rho`: 2×2 array corresponding to a mixed state.
  output Purity of a density matrix $\mathcal{P}(\rho) = \mathrm{tr}\rho^2$ (real value in $[0.5, 1.]$).

`def SvN(rho)`
input data `rho`: 2×2 array corresponding to a mixed state.
output Von Neumann entropy of a density matrix $S_{vN}(\rho) = -\mathrm{tr}(\rho \ln \rho)$ (real value in $[0., \ln 2]$).

`def entangl(psi,nspin)`
input data `psi`: 1D array corresponding to a lattice state.
`nspin`: number of spins in the lattice (integer).
output Entanglement measure on a lattice $\mathcal{E}(|\psi\rangle) = \frac{1}{N}\sum_{i=1}^{N} S_{vN}(\rho_i)$ (real value in $[0., \ln 2]$).

`def disorder(psi,nspin)`
input data `psi`: 1D array corresponding to a lattice state.
`nspin`: number of spins in the lattice (integer).
output Disorder measure on a lattice $\mathcal{D}(|\psi\rangle) = S_{vN}\left(\frac{1}{N}\sum_{i=1}^{N}\rho_i\right) - \mathcal{E}(|\psi\rangle)$ (real value in $[0., \ln 2]$).

1.4 Decomposition of the problem

In order to build simulations of the quantum control of lattice spin systems from the quantum information viewpoint, we decompose the problem in a sequence of more simple steps:

[1] By using tools from the DQCspinlattice file, we need to build the Hamiltonian of the spin system, to compute its eigenstates, and quantum information properties concerning the system. Since at the

neighbourhood of $T = 0$ K only the ground state is occupied, this state must be subject of a particular attention.

[2] By using tools of [1], we need to code a propagator of the Schrödinger equation in order to study the quantum dynamics of the spin system.

[3] By using tools of [1] and [2], we can study spectral and dynamic properties of a many-spin system with respect to its topology.

[4] Finally, by using tools of [3] we can code the control of a spin system.

Chapter 2

Hamiltonians and eigenstates of lattice spin systems

2.1 Studied models

In this chapter we want to compute and to study the eigenstates of lattice spin systems, and particularly the ground state which is the steady state of the system at very low temperature. The different models of interest are the following ones.

- Spin chains with parameters N number of spins (use $N = 8$ if not stated differently), w Larmor frequency (typically $w = 0.5$ a.u.), J exchange integral (use $J = 1$ a.u. if not stated differently), for the following different models:
  - (1) open Ising-Z spin chain,
  - (2) closed Ising-Z spin chain,
  - (3) open Ising-X spin chain,
  - (4) open Heisenberg-XXX spin chain.
- (5) Heisenberg-XYZ open spin chain with parameters: $N = 8$ number of spins, $w = 0.5$ a.u. Larmor frequency, $\vec{J} = [0.5, 1., 1.5]$ a.u. exchange integrals.
- (6) random open Ising-X chain with parameters: $N = 8$ number of spins, w_i Larmor frequency of the i-th spin as a random variable with uniform distribution into $[0., 0.5]$ a.u., $J_{i,i+1}$ exchange integral between i-th and $(i + 1)$-th spins as a random variable with uniform distribution into $[-1., 1.]$ a.u..
- (7) Spin glass with parameters: $N = 8$ number of spins, w_i Larmor frequency of the i-th spin as a random variable with uniform distribution into $[0., 0.5]$ a.u., J^u_{ij} exchange integral in u-direction between the i-th and j-th spins ($j > i$) as a random variable with uniform distribution into $[-1., 1.]$ a.u..

TP question: *Build with python the quantum Hamiltonian H as a $2^N \times 2^N$ array for each model. In the final report, give all the Hamiltonians for the cases $N = 2$, $N = 3$, and $N = 8$.*

2.2 Diagonalization algorithms

Their exists two families of algorithms to diagonalize matrices. The direct methods consist to compute the root of the characteristic polynomial. This needs a rapid method to evaluate this polynomial which exists for hermitian tridiagonal matrices by using a recurrence relation. In atomic and molecular physics, the quantum Hamiltonians become tridiagonal matrices in the numerical representation by the finite difference method. The zeros of the characteristic polynomial could be then found for example by a dichotomy method. Once known an eigenvalue λ, the associated eigenvector $|\psi\rangle$ can be computed by solving the equation $(H - \lambda \,\mathrm{id})|\psi\rangle = 0$ by using the fixed point method for example. The direct methods are efficient only on matrices with very particular structures (as tridiagonal) for which the evaluation of the characteristic

polynomial is rapid. For the general case, the computation of the characteristic polynomial at a point x by a direct computation of $\det(H - x\,\mathrm{id})$ is very slow and finally inefficient. The second family of diagonalization methods are the iterative methods which are based on the matrix power method. The more important methods are called Lanczos and Davidson methods for hermitian matrices, and Arnoldi method for non-hermitian matrices (there exist also non-hermitian adapted Lanczos and Davidson methods). In this course we are interested by these iterative methods for hermitian matrices.

2.2.1 The power method algorithm

Let H be an hermitian matrix with unknown negative spectrum $\mathrm{Sp}(H) = (\lambda_0, \lambda_1, ..., \lambda_{N-1})$ sorted by increasing values. For the sake of simplicity, we suppose that the eigenvalues are not degenerate. Let $|\psi\rangle$ a normalized nonspecial state. By nonspecial state, we mean that ψ has components onto all unknown eigenvectors $(|\phi_0\rangle, ..., |\phi_{N-1}\rangle)$: $|\psi\rangle = \sum_{i=0}^{N-1} c_i|\phi_i\rangle$ with $c_i \in \mathbb{C}$ unknown complex numbers. ψ is known in the work basis used to represent H but not in the eigenbasis. The power method algorithm is based on the following fact:

$$H^k|\psi\rangle = \sum_{i=0}^{N-1} c_i\lambda_i^k|\phi_i\rangle \simeq c_0\lambda_0^k|\phi_0\rangle \quad \textit{if } k \textit{ is very large} \tag{2.1}$$

since, being $|\lambda_0| > |\lambda_i|$ $(\forall i > 0)$, $\lim_{k\to+\infty} \frac{|\lambda_i|^k}{|\lambda_0|^k} = 0$ $(\forall i > 0)$. It follows that

$$\lim_{k\to+\infty} \frac{H^k|\psi\rangle}{\|H^k|\psi\rangle\|} = |\phi_0\rangle. \tag{2.2}$$

We can then find the ground state of H by studying the sequence $\left(\frac{H^k|\psi\rangle}{\|H^k|\psi\rangle\|}\right)_{k\in\mathbb{N}}$. The ground level is found as $\lambda_0 = \langle\phi_0|H|\phi_0\rangle$ with $|\phi_0\rangle$ the limit of the sequence.

If the spectrum of H is not negative, we simply use the method for a shifted matrix $H' = H - \mu\,\mathrm{id}$ where $\mu > 0$ is sufficiently large such that the spectrum of H' be negative. H and H' have the same eigenvectors and $\mathrm{Sp}(H) = \mathrm{Sp}(H') + \mu$.

If λ_0 is degenerate, the algorithm converges to an arbitrary state (which depends on the choice of $|\psi\rangle$) into the eigensubspace associated with λ_0.

To find the second eigenvector, we apply the same method but with a more special state $|\psi_0\rangle$ which has no component onto $|\phi_0\rangle$ but has components onto all other eigenstates. In practice $|\psi_0\rangle = (1 - |\phi_0\rangle\langle\phi_0|)|\psi\rangle$. We have then

$$\lim_{k\to+\infty} \frac{H^k|\psi_0\rangle}{\|H^k|\psi_0\rangle\|} = |\phi_1\rangle \tag{2.3}$$

and so on for the other eigenvectors.

The pseudo-code of the power method algorithm for the ground eigenvector is the following:

---

take random vector $|\phi_0\rangle$
normalize $|\phi_0\rangle$
$H \leftarrow H - shift * \mathrm{id}_{2^N}$
while $\||H|\phi_0\rangle - \langle\phi_0|H|\phi_0\rangle|\phi_0\rangle\| > \epsilon$ and $k \le k_{\max}$ do
    $|\phi_0\rangle \leftarrow H|\phi_0\rangle$
    normalize $|\phi_0\rangle$
    $k \leftarrow k + 1$
end while
$H \leftarrow H + shift * \mathrm{id}_{2^N}$
$\lambda_0 \leftarrow \langle\phi_0|H|\phi_0\rangle$

The parameters of the algorithm are

- $shift$ (positive real number): the shifting value used to ensure a negative spectrum;
- ϵ: the wanted precision concerning the verification of the eigenequation (typically $\epsilon = 10^{-8}$);
- $k_{\max}$: the maximal number of iterations, if k reaches $k_{\max}$ before the eigenequation be satisfied with the precision ϵ then the algorithm has not converged.

For the first excited eigenstate the pseudo-code is the following:

---

take random vector $|\phi_1\rangle$
$|\phi_1\rangle \leftarrow |\phi_1\rangle - \langle\phi_0|\phi_1\rangle|\phi_0\rangle$
normalize $|\phi_1\rangle$
$H \leftarrow H - shift * \mathrm{id}_{2^N}$
while $\||H|\phi_1\rangle - \langle\phi_1|H|\phi_1\rangle|\phi_1\rangle\| > \epsilon$ and $k < k_{\max}$ do
    $\phi_1 \leftarrow H\phi_1$
    $|\phi_1\rangle \leftarrow |\phi_1\rangle - \langle\phi_0|\phi_1\rangle|\phi_0\rangle$
    normalize $|\phi_1\rangle$
    $k \leftarrow k + 1$
end while
$H \leftarrow H + shift * \mathrm{id}_{2^N}$
$\lambda_1 \leftarrow \langle\phi_1|H|\phi_1\rangle$

Note that we project onto the orthogonal complement of $|\phi_0\rangle$ at each iteration (and not only at the beginning) because non-zero components onto $|\phi_0\rangle$ appears into $|\phi_1\rangle$ due to the numerical errors.

TP question: *By using the power method, code a python program computing the two first eigenvalues and the associated eigenvectors of H (write the eigenvectors in the Dirac notation in the canonical basis, by rounding the various coefficients to two decimal places). In the final report, only for model (3) compare the results obtained with your code with the ones given by the python function* `la.eigh` *(compare the precisions in the validity of the eigenequation and search the number of iterations needed to obtain the same precision).*

2.3 Properties of the ground state

2.3.1 Para- and ferromagnetic systems

TP question: *For the models (2), (3) and (6) of Sec.* 2.1*: compute the ground state $|\phi_0\rangle$, $S_{vN}(\langle\rho\rangle)$, $\mathcal{E}(|\phi_0\rangle)$ and $\mathcal{D}(|\phi_0\rangle)$; plot on the same graph the populations, the modules of the coherences, and the von Neumann entropies with respect to the spins onto the lattice. Compare and comment the results.*

Hints: make your simulation with an increasing number of spins in order to debug your code. It can be interesting to plot graphs as a function of the system parameters, such as $N, \omega_i, \vec{J}_{i,j}$. However, in the final report analyze only the cases asked above.

2.3.2 Antiferromagnetic systems

TP question: *Compute the ground states of a ferromagnetic Ising-Z open chain of $N = 8$ spins (with $w = 0.$, $J = 1.$ a.u.) and of an antiferromagnetic Ising-Z open chain of $N = 8$ spins (with $w = 0.$, $J = -1.$ a.u.). Compare the ground state of an antiferromagnetic material with the one of a ferromagnetic material. How are the spins organized in the two cases?*

TP question: *Compute the ground states of a ferromagnetic Ising-Z closed chain of $N = 3$ spins (with $w = 0.$, $J = 1.$ a.u.) and of an antiferromagnetic Ising-Z closed chain of $N = 3$ spins (with $w = 0.$, $J = -1.$ a.u.). These chains can be interpreted as one cell of triangular lattices. Compare the ground states of these two spin triangles. What is the difference with the case of the open spin chains?*

An antiferromagnetic spin triangle is said to exhibit a geometric frustration phenomenon.
TP question: *From your observations and/or personal research in the literature, explain this notion.*

Remark: Spin ices also present geometric frustrations but these induce other magnetic phenomenons (which are not the subject of the present course). These structures are called spin ice because the organization of the spins in the ground state is the same than the organization of the electric dipolar momenta in water ice (we recall that the H_2O molecule is highly polarized).

Chapter 3

Dynamics of lattice spin systems

3.1 Studied models

In this chapter we study the quantum dynamics of a spin chain, and more precisely the dynamics of an excitation in the chain. The restriction of the study to chains is just a practical choice to have simple graphical representations. The models we are interested to are the following:

- (1) Heisenberg-XXX open chain of $N = 7$ spins with $w = 1.$ a.u. (Larmor frequency) and $J = 0.1$ a.u..
- (2) Heisenberg-XXX closed chain of $N = 7$ spins with $w = 1.$ a.u. and $J = 0.1$ a.u..
- (3) inhomogenous Heisenberg-XXX open chain of $N = 7$ spins with $w = 1.$ a.u. and with $J_{i,i+1} = \frac{0.5}{i}$ a.u. [exchange integral between the i-th and the $(i+1)$-th spins].

TP question: *For each model, build with python the quantum Hamiltonian* `H` *as a* $2^N \times 2^N$ *array.*

3.2 Spectral integrator

The study of the dynamics of the spin chain needs the integration of the Schrödinger equation

$$\mathbb{i}\frac{d|\psi(t)\rangle}{dt} = H|\psi(t)\rangle, \tag{3.1}$$

with H a time-independent Hamiltonian. There exist two kinds of numerical integrators of the Schrödinger equation: the finite difference integrators (as the Richardson algorithm) and the unitary integrators (as the DVR/FBR split operator algorithm). Unitary integrators are similar to the symplectic integrators used in classical dynamics (see the dynamical systems M1 course). They always preserve the normalization of the quantum state (as the symplectic integrators always preserve the classical energy). In this course we are interested by particular cases of unitary integrators called spectral integrators. These ones are based on the diagonalization of the Hamiltonian and use the linearity of the Schrödinger equation. For time-independent Hamiltonians, the spectral integrator is universal in the sense that all spectral integrator schemes are reduced to this one in the time-independent case. The method can be extended to time dependent Hamiltonians (using a split-operator scheme, for example).

Let $|\psi_0\rangle$ be the normalized initial state, $\mathrm{Sp}(H) = (\lambda_i)_i$ be the spectrum of H, and $(|\phi_i\rangle)_i$ be the associated normalized eigenbasis. Since H is time-independent, the evolution operator is a simple matrix exponential:

$$U(t,0) = e^{-\mathbb{i}Ht} = \sum_i e^{-\mathbb{i}\lambda_i t}|\phi_i\rangle\langle\phi_i|. \tag{3.2}$$

It follows that

$$|\psi(t)\rangle = U(t,0)|\psi_0\rangle = \sum_i \langle\phi_i|\psi_0\rangle e^{-\mathbb{i}\lambda_i t}|\phi_i\rangle. \tag{3.3}$$

The spectral integrator is then very simple, it consists to diagonalize H and to apply the previous formula.

TP question: *Define a python function corresponding to the spectral integrator with the following requirement:*

`def Dyn(`H`,t,psi0)`

input data H: 2D array corresponding to the quantum Hamiltonian represented in any basis, or 1D array corresponding to the diagonal of the quantum Hamiltonian in its eigenbasis.
`t`: duration of the time propagation (real number).
`psi0`: 1D array corresponding to the initial quantum state represented in the same basis than H.

output 1D array corresponding to $|\psi(t)\rangle = U(t,0)|\psi_0\rangle$ represented in the same basis than H.

3.3 Dynamics

We consider a spin chain with initial state $|\psi_0\rangle = |0001000\rangle$ or $|\psi_0\rangle = |1000000\rangle$ which correspond to the cases where almost all spins are in their individual ground state but with an excitation onto the site 4 or 1.

TP question: *For each model (1) to (3) of Sec* 3.1 *integrate the Schrödinger equation on the time interval* $t \in [0, 500]$ a.u. *and compute a list of lists:*

$$\texttt{pop} = [[(\rho_k(i\Delta t))_{11} \text{ for } k \in \{1,7\}] \text{ for } i \in \{0, N_{\text{time}}\}],$$

where $N_{\text{time}} = 200$ *and* $\Delta t = \frac{T}{N_{\text{time}}}$. $\rho_k(t)$ *is the reduced density matrix of the k-th spin at time t, and then* $(\rho_k(t))_{11}$ *is the population of the state* $|1\rangle$ *for the k-th spin at time t.*
For each model and for the two above initial conditions plot a density graph of the population of the state $|1\rangle$ *with respect to the spins of the chain (abscissae) and to the time (ordinates), with a color gradient associated with the occupation probability of* $|1\rangle$ *(in* $[0, 1]$*).*

Such a graph can be obtained with the `pyplot` function `contourf` as

```
figpop, graphpop = py.subplots();
gp=graphpop.contourf(range(1,nspin+1),[i*deltat for in range(Ntime)],pop,[i*0.02 for in
range(51)],cmap='hot',antialiased=True);
graphpop.set_xlabel('spins');
graphpop.set_ylabel('$t$');
figpop.colorbar(gp);
figpop.savefig('graphpop.png');
```

TP question: *Comment the graphs. Explain the behaviours of the quantum excitation. What is the important difference between open and closed chains? For the model (1) let vary w and J (using also smaller and larger values than the ones used before), to establish how the propagation of a quantum excitation in the spin chain depends on the Larmor frequency and the exchange integral (study in particular the propagation speed of the quantum excitation).*