Finite temperature 2D spin lattices solved with minimally entangled typically thermal states (METTS) algorithm

Javier Robledo Moreno
Supervised by Miles Stoudenmire (CCP Flatiron Institute)
Computationa Physics

December 21, 2018

1 Introduction

The goal of this project was to analyze the magnetic properties of antiferromagnet materials using minimally entangled typically thermal states (METTS) algorithm [1]. Recent years experiments have shown remarkable properties of antiferromagnet materials with certain geometries. Specifically, magnetization curves (magnetization as a function of the external applied field at a fixed temperature) show regions in which the magnetization stays constant over a range of applied magnetic field at sufficiently low temperatures (see figure 1). Previous theoretical and computational studies have shown that the plateau formation occurs when the system reaches a highly ordered phase induced by the long-range order of the system [3, 4, 5]. These studies used Quantum Monte Carlo (QMC) techniques or exact diagonalization. However, METTS algorithm has never been used to address this question. Therefore, the goal of this project was to implement METTS algorithm and test whether it is adequate or not to describe this physical situation.

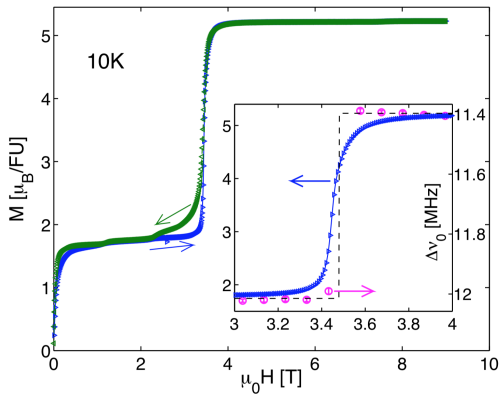


Figure 1: Measured magnetization curve for $Ca_3Co_2O_6$ (effectively a 2D triangular lattice geometry) from reference [2]. Note that temperature is fairly low ($T = 10K$). Inset is not relevant for the purpose of this project.

In order to obtain magnetization curves, the magnetization of the system was computed using the following definition:

$$\langle M \rangle = \frac{1}{SN} \sum_{i=1}^{N} \langle S_i^z \rangle, \tag{1}$$

where N is the number of spins in the system and S is the value of the spin that each particle carries. In this case, $S = 1/2$ spins were studied. The Heisenberg model for spins in a lattice was used to describe the magnetic behaviour of an antiferromagnetic material as it has been proven to accurately describe magnetic systems. The Hamiltonian of the system under the Heisenberg description can then be written as:

$$H = \sum_{\langle i,j \rangle} J_{i,j} \left(S_i^x S_j^x + S_i^y S_j^y + S_i^z S_j^z \right) - h \sum_i S_i^z, \tag{2}$$

where $\vec{S_i}$ are spin S operators acting on site i and $J_{i,j}$ are the exchange constants between a given $\langle i, j \rangle$ pair in the lattice. For the model to describe the behaviour of antiferromagnets, $J_{i,j}$ was chosen to be positive. The dimensionless version of the model that was used in the computations is: $J_{i,j} \to J_{i,j}/J_{i,j}$, $h \to h/J_{i,j}$, $k_B T \to k_B T/J_{i,j}$.

As mentioned before, previous studies have used QMC or exact diagonalization to solve this system [3]. However, the problem with QMC is that it presents the fermionic sign problem, making the size of error bars of average quantities to be fairly large [3, 4, 5]. In a nutshell, the fermionic sign problem is a numerical sign problem that arises from the anti-symmetry of the wave function of fermionic systems. The problem associated with the exact diagonalization of the system, is that the size of the problem grows exponentially with the number of spins. This is where METTS algorithm becomes extremely useful. It will allow to randomly sample the configuration space as QMC but without presenting a fermionic sign problem [1].

This report is structured as follows: in section 2, I describe METTS algorithm, first, by outlining the basic idea, to later give a detailed description of each step. In this section I also give a brief introduction about tensor networks, as they will be used to illustrate certain steps of the algorithm. To conclude this section I describe how does the run-time of the algorithm scales with the different parameters of the simulation and why the algorithm is called minimally entangled typically thermal states. In section 3, I show the magnetization curves obtained with this algorithm for different lattice geometries and I will explain the origin of the highly ordered phases. In section 4, I discuss the conclusion of this study and the future plans I have to keep working in this matter. Finally, section 5 describes the implementation of the algorithm and the code submitted with this report.

2 METTS algorithm

As described before, we are interested in measuring expectation values of an operator within the thermodynamical canonical ensemble. The expectation value of operator $\hat{A}$ at temperature $T \Rightarrow \beta = 1/k_B T$ can be written as:

$$\langle \hat{A} \rangle = \frac{1}{Z} \operatorname{Tr} \left[e^{-\beta H} \hat{A} \right], \tag{3}$$

where Z is the partition function.

The direct, brute force computation of this expectation value has two mayor difficulties. The first and most clear one is that the size of the matrices scale exponentially with the number of spins, making even matrix multiplication intractable on a regular computer for systems with more than fourteen spins. There is also the problem of exponentiating the Hamiltonian, which in general is computationally heavy as it involves many matrix multiplications because one expands the exponential as a power series.

2.1 Intuitive description

One possible way of going around the issues associated with exact diagonalization and QMC is to conveniently rewrite the wanted expectation value in equation 3 as:

$$\langle \hat{A} \rangle = \frac{1}{Z} \operatorname{Tr} \left[e^{-\beta H/2} \hat{A} e^{-\beta H/2} \right] = \frac{1}{Z} \sum_i \langle i | e^{-\beta H/2} \, \hat{A} \, e^{-\beta H/2} | i \rangle, \tag{4}$$

where $\{|i\rangle\}$ is just a basis of our choice. The next step is to define a set of normalized states $|\phi(i)\rangle$ as:

$$|\phi(i)\rangle = \frac{1}{\sqrt{P(i)}} \, e^{-\beta H/2} \, |i\rangle, \tag{5}$$

where the normalization factor takes the value: $P(i) = \langle i|e^{-\beta H}|i\rangle$. With these definitions it is possible to rewrite equation 4 as:

$$\langle \hat{A} \rangle = \sum_i \frac{P(i)}{Z} \langle \phi(i)|\hat{A}|\phi(i)\rangle. \tag{6}$$

The goal of METTS algorithm is to efficiently sample states $|\phi(i)\rangle$ with probability $P(i)/Z$ so we can compute the expectation value of any operator with the right probability distribution, as shown in equation 6.

As a final comment, it is important to remark that, as described above, one can choose any basis $\{|i\rangle\}$ to expand the trace, however, it is convenient to choose the base of classical product states (CPS $|i\rangle = |i_1\rangle|i_2\rangle...|i_N\rangle$) as they are states with no entanglement. This fact will make the computation significantly faster.

2.2 Basics on tensor networks

A detailed explanation of the algorithm will require a lot of tensor algebra. An easier way of visualizing these calculations is to use tensor networks, for this reason, before jumping into the description of the algorithm, some tensor network notation and definitions will be introduced. Figure 2 contains all the tensor network definitions needed to understand the different steps of the algorithm. Without getting into the mathematical details and definitions, a tensor is an object (a geometrical shape as a square, circle... in the tensor network representation) with indexes (legs in tensor network representation). For example, a vector and a matrix are shown in box a).

It is also possible to visualize tensor operations in this graphical representation. Connecting two legs means to contract those indexes. Some examples are shown on box b). This representation becomes especially useful when representing many-body wave functions and operators and their operations, as shown in boxes c), d) and e).

2.3 Detailed explanation of the algorithm

In this section, a detailed explanation of all the steps in the algorithm will be given. All the steps are depicted in the diagram in figure 3.

- **STEP 0:** Start by choosing a random CPS. Usually a Neel state is used to initialize the computation ($|i\rangle = |\uparrow\rangle|\downarrow\rangle|\uparrow\rangle|\downarrow\rangle...$). After describing all the steps of the algorithm I will describe what has to be done in order to not to influence the computation with the choice of the initial CPS.

- **STEP 1:** Write the CPS as a matrix product state (MPS). The notion of MPS is just another way of writing the wave function:

$$|\psi\rangle = \sum_{i_1,i_2,..,i_N} \psi_{i_1,i_2,..,i_N}|i_1\rangle|i_2\rangle...|i_N\rangle = \sum_{\{s\}} A^{s_1}A^{s_1}A^{s_2}...A^{s_N}|s_1\rangle|s_2\rangle...|s_N\rangle, \tag{7}$$

  where s_i label local basis elements and A^{s_i} is a matrix. The tensor network representation of this transformation is shown in figure 4. Another very important transformation for this algorithm is shown in figure 4 b), the singular value decomposition (SVD). The SVD of any matrix $\hat{A}$ (rank two tensor of size $M \times N$) is to write it as the product of three matrices:

$$\hat{A} = UDV^\dagger, \tag{8}$$

  where U and $V^\dagger$ are unitary matrices of size $M \times M$ and $N \times N$ respectively and D is a diagonal matrix of size $M \times N$. As U and $V^\dagger$ are unitary matrices, the satisfy the following orthogonality relation: $UU^\dagger = I$ and $VV^\dagger = I$, which will be a really important property for the efficiency of the algorithm.

  The last thing to note about SVD's, is that the diagonal matrix can be contracted with either U or $V^\dagger$ as shown in figure 4 b), where it is contracted with U to form a MPS. The site chosen to

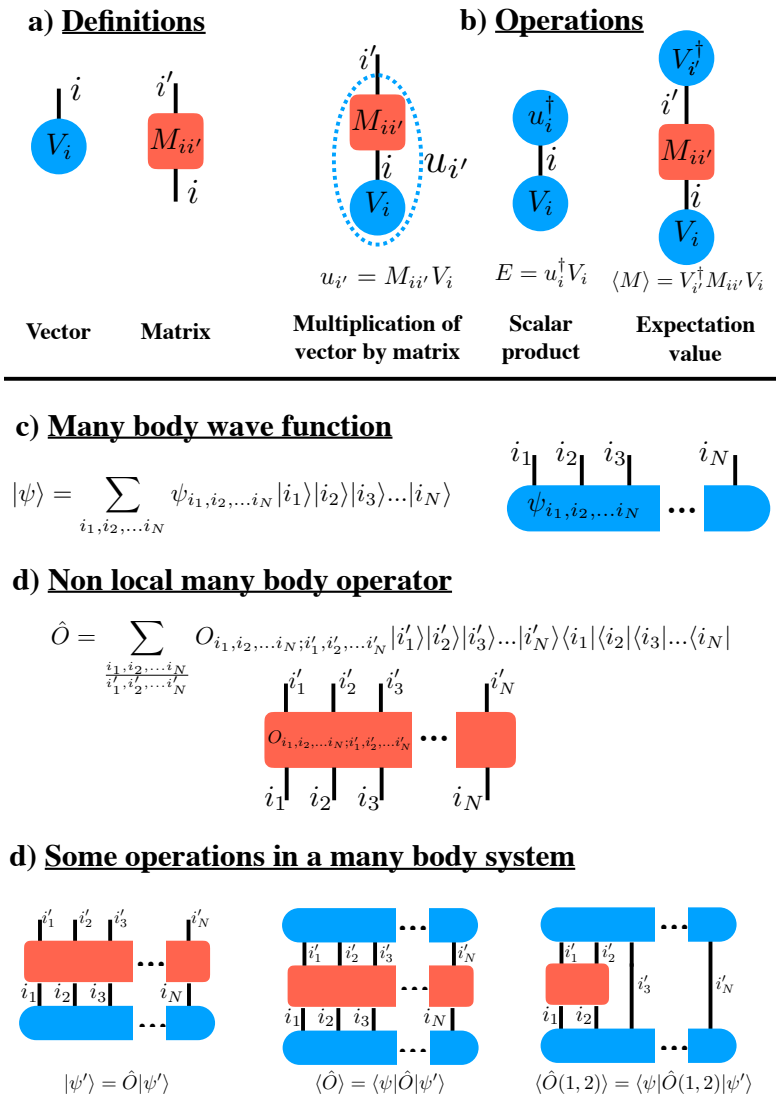


Figure 2: Tensor network definitions and conventions. **a)** Convention to represent a vector and a matrix. **b)** Representation of basic operations, contracting indexes, matrix multiplication and scalar product. **c)** Representation of a many-body wave function (using the tensor product representation). **d)** Non local many-body operator. **e)** From left to right: Non local many body operator acting on a wave function, expectation value of a non local operator and expectation value of a local operator.

contract the diagonal matrix will be called orthogonality center (left site in figure's 4 example). In this report, orthogonality centers will be distinguished from regular sites in the tensor by coloring them in red.

Even though 4 b) only shows an example of how to apply the SVD to a wave function of two particles (two indexes), this idea can be generalized to an arbitrarily big wave functions by performing SVD decompositions to each pair of consecutive indexes and move the orthogonality center to the desired position.

- **STEP 2:** Construct the METTS. As seeing in formula 5, in order to obtain a METTS from a CPS, we need to construct the operator $e^{-\beta H/2}$. A brute force approach to doing this computation would scale exponentially with the number of spins, thus this option is discarded. To exponentiate the Hamiltonian, the Trotter-Suzuki decomposition will be used. If the Hamiltonian can be written as the sum of Hamiltonians acting in smaller parts of the system ($H = \sum_n H_n$, where

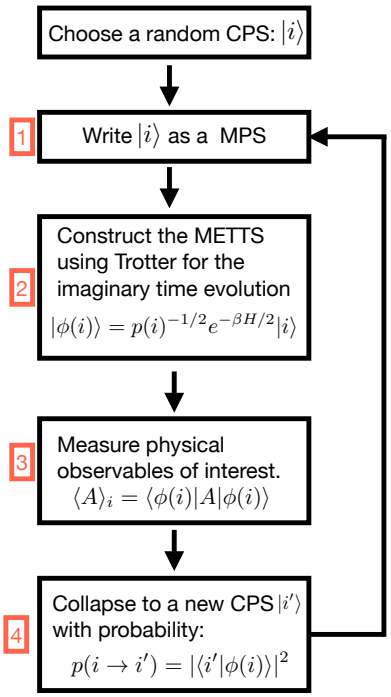


Figure 3: Diagram with all the steps of METTS algorithm. Boxes represent the different steps explained in section 2.3.

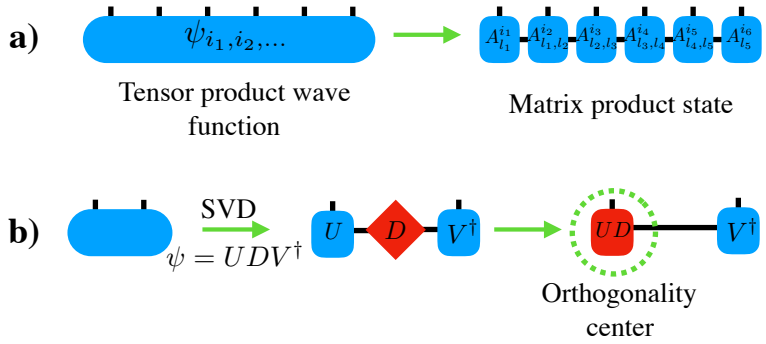


Figure 4: **a)** Tensor network representation of a tensor product wave function (left) and MPS representation of the same state (right). **b)** Singular value decomposition (SVD) of two indexes of a wave function as a possible transformation to write an arbitrary state as a MPS

$[H_i,\ H_j] \neq 0$ in general) then [7]:

$$e^{-\beta H/2} \simeq e^{-\beta H_1/4} e^{-\beta H_2/4} \ldots e^{-\beta H_N/2} \ldots e^{-\beta H_2/4} e^{-\beta H_1/4} + O\left(\left(\frac{\beta}{2}\right)^3\right). \tag{9}$$

This decomposition makes the problem tractable as in our case we can write the Hamiltonian as a sum of Hamiltonians acting in pairs of particles (see eq. 2). Therefore, the dimension of each exponential term in equation 9 is 4 (4×4 matrix). Exponentiation of a small matrix can be done efficiently and exactly up to rounding error. To do so, all we have to do is to diagonalize the matrix, exponentiate the eigenvalues and apply a rotation to go back to the non diagonal basis. Another way of doing this exponentiation for small matrices is to decompose the exponential as a Taylor series and cut it at a certain power. However this second method is not as accurate and can be slower if we are interested in high accuracy.

A very important step to keep the error of the Trotter decomposition bounded, is to move the orthogonality centre to one of the sites in which $e^{-\beta H_i/4}$ is acting. Even though each exponential term is tried to be constructed with the minimum error possible as described above,

it will always carry some error associated with its construction. Moving the orthogonality center to one of the sites in which the exponential term is applied will prevent the error to spread across the MPS. One way to measure the error in an element $|\psi_{computed}\rangle$ of a vector space with respect of the exact value it should have $|\psi_{exact}\rangle$, is to define the equivalent of an L_1 norm as $L_1 = 1 - \langle\psi_{exact}|\psi_{computed}\rangle$. If $|\psi_{computed}\rangle = |\psi_{exact}\rangle$ then $L_1 = 0$, whereas when the two vectors are as far as possible from each other (anti-parallel) then $L_1 = 2$. Coming back to the error associated to applying an exponential term to an MPS, according to the definition of the error provided above, if we move the orthogonality centre as described, then the only part that will contribute to the scalar product will be the part associated with the indexes where we applied the operator and not the entire wave function, avoiding the error to spread across the MPS.

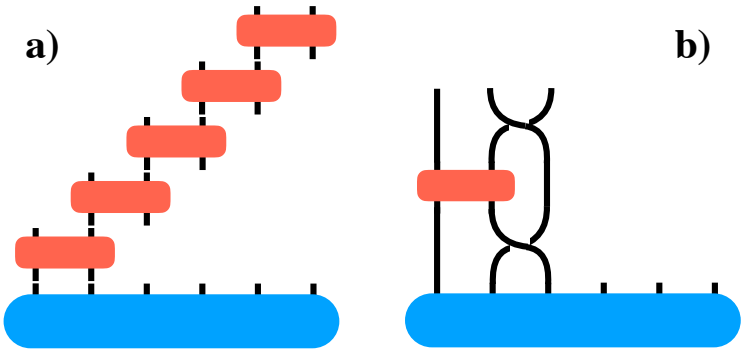


Figure 5: **a)** Tensor network representation of the Trotter-Suzuki decomposition when the Hamiltonian has only nearest neighbour interactions in a 1D geometry. The red boxes represent the exponential terms in the right hand side of equation 9. **b)** Tensor network representation of the swap of two sites, required when the Hamiltonian involves interactions between non neighbouring sites.

The tensor network representation of Trotter's decomposition is shown in figure 5 a), where we can see that the calculation is only directly applicable in systems with only nearest neighbour interactions in 1D arrangements of the spins. In order to be able to apply exponential terms involving non neighbouring spins, we would first have to apply a swap gate, which exchanges the position of two of the indexes, apply the exponential term in the now consecutive indexes and undo the swap of the indexes. This trick is represented in figure 5 b). The swap gate, exchanging two sites of indexes i and j can be written as $S(i,j) = \delta_{s_i,s'_j}\delta_{s_j,s'_i}$. The swap gate will be used in almost all the physical systems studied, as the magnetization plateaus we are interested in, appear when there is longer-range interactions.

One last, but really important comment about the Trotter decomposition is that the error associated with it is of order $(\beta/2)^3$, which is a good scaling of the error when $\beta << 1 \Rightarrow$high temperatures. However, the plateau phenomenon we are interested in, only occurs at very low temperatures, and therefore, high values of β, meaning that, in principle, the Trotter decomposition would be a bad approach to the problem. However, it is possible to come around this issue by rewriting $\beta = m\tau$ where m is an integer and $\tau << 1$. Therefore, the exponential of the Hamiltonian at inverse temperature β can be written as the periodic application of the exponential of the Hamiltonian at inverse temperature τ, m times:

$$e^{-\beta H/2} = \left(e^{-\tau H/2}\right)^m. \tag{10}$$

Now the error was reduced from $O((\beta/2)^3) = O((m\tau/2)^3)$ to $O(m\cdot(\tau/2)^3)$ which is a very big improvement. Without this reduction in error, it would not be possible to use the Trotter decomposition for low temperatures.

- **STEP 3:** Measuring the expectation value of local operators. In particular, we are interested in measuring the magnetization of the system as defined in equation 1 which is the sum of local expectation values. Once again, doing a brute force calculation, would require exponential

time and memory. That is why the notion of orthogonality centre will be used to shorten the computation time required.

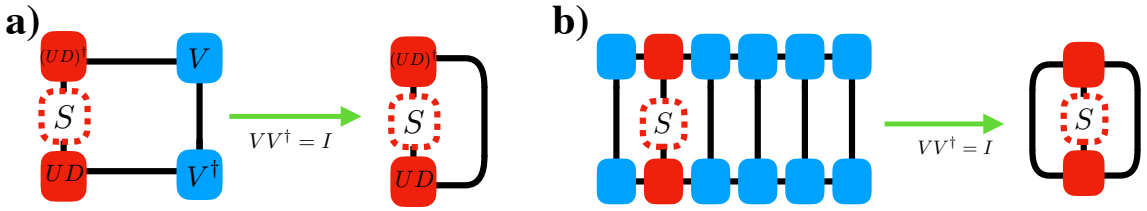


Figure 6: Demonstration of how the notion of orthogonality center makes the measurement problem tractable. **a)** Example in a wave function for two particles. It shows how contracting the indexes for the non orthogonality centre site reduces the problem to computing the expectation value only in the orthogonality centre. **b)** Generalization to a wave function of more than two particles, where again the problem is reduced to just computing the expectation value in the orthogonality centre.

The steps to measure the expectation value of the spin operator in the z direction in position i $\langle\phi|S_i^z|\phi\rangle$, are the following: first, the orthogonality centre will be moved to position i using SVD's (as shown in figure 6 a) and b) in red) so the rest of the MPS will be orthogonal to itself ($VV^\dagger = I$). As shown in figure 6 a) and b), when taking the expectation value of S_i^z, as the orthogonality centre was chosen to be on site i, then the problem reduces to just the computation of the expectation value on site i and not the rest of the wave function. This step is crucial because it makes the computation of the magnetization to happen in polynomial time instead of exponential time.

- **STEP 4:** Collapse the MEETS $|\phi(i)\rangle$ to a new CPS $|i'\rangle$ with probability $P(i \to i') = |\langle i'|\phi(i)\rangle|^2$. This step ensures that the METTS are being sampled with the right probability given in equation 6. Imagine that we have the ensemble of CPS initially distributed with probability $P(i)/Z$. Now if we randomly choose a CPS from this ensemble and and follow steps 0 to 4, the probability of measuring a particular CPS $|i'\rangle$ at the end of the collapsing state is:

$$\begin{aligned}\sum_i \frac{P(i)}{Z} P(i \to i') &= \sum_i \frac{P(i)}{Z} |\langle i'|\phi(i)\rangle|^2 \\ &= \sum_i \frac{P(i)}{Z} \frac{|\langle i'|e^{-\beta H/2}|i\rangle|^2}{P(i)} \\ &= \langle i'|e^{-\beta H}|i'\rangle/Z \\ &= P(i')/Z,\end{aligned}$$

which is the desired result to ensure that expectation values are being computed properly. Therefore, the ensemble of CPS with the probability distribution $P(i)/Z$ is a requirement for this algorithm.

Let's analyze a very important feature of the algorithm. One might have noticed that in this sampling scheme, when one collapses to a new CPS, this new CPS depends on the previous CPS. In particular, it is easy to see that the first few METTS and CPS generated will be correlated with the initial CPS, which is something we want to avoid when computing physical quantities. Therefore, we have to do a few iterations of this algorithm in order for the METTS generated to be completely independent of the first CPS. This is a general feature of random sampling algorithms such as Monte Carlo algorithms [8]. The number of iterations that it takes for the simulation to generate samples not correlated with the initial state is called the autocorrelation time.

The choice of the CPS basis in STEP 4 is, in principle, not relevant to get the correct expectation values from this algorithm. However, there are certain advantages of choosing certain CPS basis. One might be tempted to always collapse along the z as the projectors are diagonal in the chosen basis, but

collapsing along just one direction produces strong correlation effects and thus long autocorrelation times [1]. Avoiding long autocorrelation times is key for reducing the run-time drastically as the iterations withing the autocorrelation time cannot be used to compute physical quantities. The way of solving this issue is to change the axis of projection between consecutive iterations of the algorithm. Specifically, in the implementation that I did, I alternated collapsing to the z and x directions.

2.4 Run-time

It has been said in the description of the algorithm that the ultimate goal of each step was to make the run-time not to scale exponentially with the size of the system, however, it was not specify how the run time scales with the size of the system on each case. In this section this scale will be specified more on detail. Before starting with this description, it is important to define the bond dimension of an MPS. When writing a wave function as a MPS, one associates a tensor with three indexes to each site of the wave function tensor (see figure 4 a)). Only one of those three indexes is free and the other two are contracted with the tensors on each side (bond indexes). The bond dimension b is defined as the largest dimension out of all the bond indexes of the wave function. As a general rule, more entangled states will have larger values of the bond dimension which translates to longer run-times.

The run-time scaling of the algorithm steps presented above is the following:

- Cost of producing a METTS: the cost scales as $O(N_x N_y^2 \beta b^3)$, where N_x and N_y is the size of the lattice in x and y directions where the spins are sitting. This scaling comes from having to perform $N_x \cdot N_y^2$ SVD's to move the orthogonality centre to each position of the lattice where the local Hamiltonians are acting. The cost of computing a SVD scales as b^3 [8]. The factor β comes from the fact that we have to periodically apply $\exp(-\tau H/2)$ m times at low temperatures.

- Cost of measuring the magnetization: the cost scales as $O(N_x N_y b^3 d^2)$ where d is the size of the local Hilbert space (for a 1/2 spin system $d = 2$). Once again the term, $N_x N_y b^3$ comes from having to perform $N_x \cdot N_y$ SVD's in the lattice, while the factor d^2 comes from the matrix multiplication with S_i^z that has size $d \times d$.

- Cost of collapsing to a new CPS: the cost scales as $O(N_x N_y b^3 d^2)$. Once again, as when measuring the magnetization, to compute the probability of collapsing we have to compute the expectation value of projectors acting on one site of size $d \times d$ in every site.

To end this section I would like to make to remark that, as it follows from the description of the algorithm, it is embarrassingly parallel. This feature was exploited co collect data more efficiently.

2.5 Why is $|\psi(i)\rangle = e^{-\beta H/2}|i\rangle$ minimally entangled?

The reason it is said that they are minimally entangled is that they are constructed from product states that are not entangled, therefore it is reasonable that the entanglement entropy of these new states to be minimum. Moreover, it can be proven that METTS have all the properties expected from a thermal system. Specifically, at high temperatures they behave like classical systems while at low temperatures they show a very quantum behaviour with spontaneous symmetry breaking of certain Hamiltonians. They also stay factorized over parts of the system that do not interact as thermal states do [1].

3 Application to physical systems

The algorithm described above was used to compute magnetization curves (magnetization as a function of the magnetic field applied for a fixed temperature) for spins in a lattice. The ultimate goal was to compute the magnetization curve at low temperature for a 2D triangular lattice as there are many experimental realizations of this kind of geometry [2].

To become familiar with the algorithm and to test that everything was implemented correctly, I first studied the simplest geometry possible: a 1D chain of spins with only nearest neighbour interactions

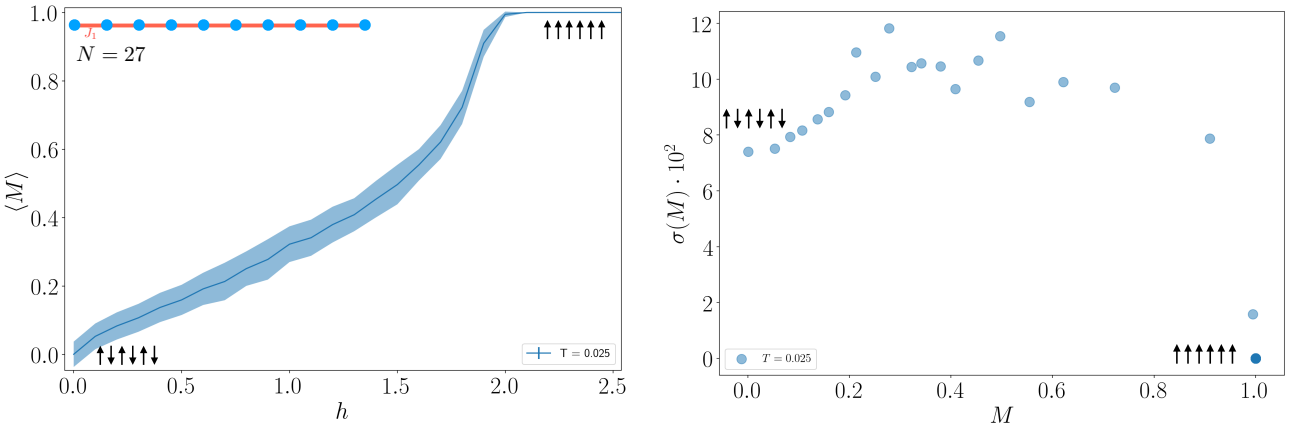
(see inset of figure 7a). This will also serve as a toy model to describe the relevance of thermal fluctuations in the study of ordered phases. However, in this geometry, no plateaus are expected because in 1D no spontaneous symmetry breaking can occur [9].

The next natural step would seem to be studying a 1D chain with next-nearest neighbour interactions, however this is effectively a triangular lattice stretched in the x direction with $N_y = 2$ (see inset of figure 9a). This makes that every site on the lattice to effectively have 4 nearest neighbours. The issue with increasing the number of nearest neighbours is that the entanglement becomes larger and the bond dimensions of the MPS becomes larger, making the computation much slower. Therefore, there is a simpler geometry that can be studied and that has some physical realizations: the dimer chain (see inset of figure 8a). In this geometry every site of the lattice is only connected with 3 nearest neighbours, making the computation significantly faster than for the 1D chain with next-nearest neighbours. Furthermore, the magnetization structure of the dimer chain is extremely interesting as it shows an infinity number of plateaus at zero temperature [3]. After computing the dimer chain, the 1D chain with next-nearest neighbours was also studied.

Finally, the triangular lattice will be computed. The issue with this computation is that it becomes significantly slower as now every site of the lattice has 6 nearest neighbours, making the entanglement and therefore the bond dimension much larger than in the previous cases. In this case the use of computer clusters running over the course of weeks is required. Periodic boundary conditions in the y direction were used.

3.1 1D nearest neighbour chain

In this case we used a value for the exchange interaction $J_1 = 1$ and a inverse temperature of $\beta = 40$. In this case we are not interested in going to lower temperatures as it is known that this geometry does not present plateaus not even at zero temperature [3] so we chose a higher temperature to make the simulation time shorter. The results for magnetization curve and fluctuations are shown in figure 7.



(a) Magnetization as a function of the applied field. Inset in the top left shows the geometry studied.

(b) Temperature fluctuations of the magnetization (from figure 7a) as a function of the magnetization.

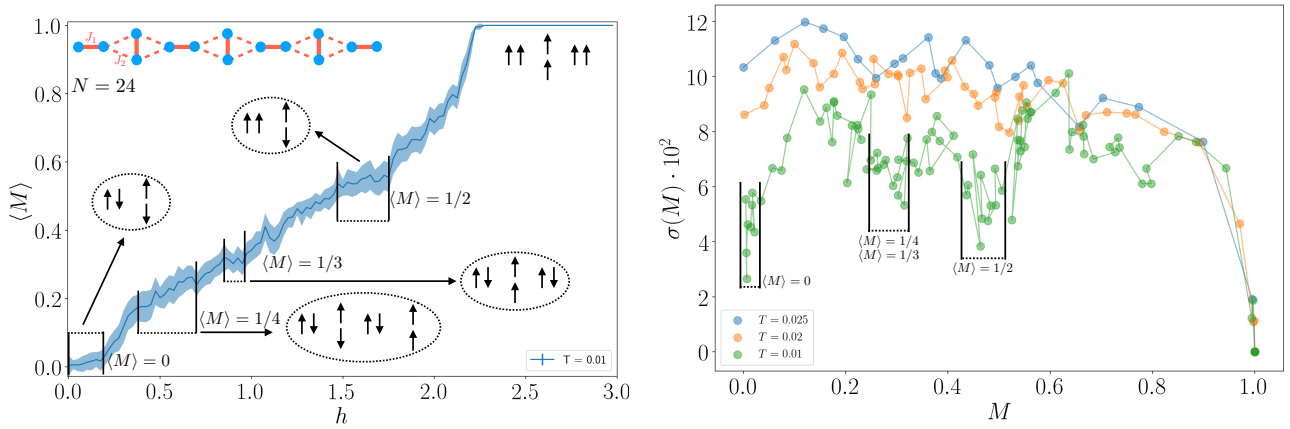
Figure 7: Magnetization curve and temperature fluctuations of the magnetization $\sigma(M)$ for a 1D chain of $N = 27$ spins with only nearest neighbour interactions at inverse temperature $\beta = 40$ averaged over 100 samples. The solid line shows the average and the shadowed region represents the fluctuations around the average values due to temperature fluctuations. Arrows show the spin orientation at highly ordered phases.

Figure 7a shows how magnetization grows with the applied field as expected for an antiferromagnet. At zero applied field the spins are in a Neel ground state (anti-aligned with their nearest neighbours) showing a zero value for the magnetization. As field increases, the field term in the Hamiltonian competes with the antiferromagnetic interaction making the spins to align with the external field. When finally all the spins are aligned with the magnetic field, the magnetization reaches its saturation value. This is a very standard behaviour for a magnetic material.

A very interesting property of the systems to analyze, is the magnetization thermal fluctuations, which are shown in figure 7b as a function of the magnetization. We can observe that there are two values of magnetization for which the fluctuations are smaller than for the rest of the values ($\langle M \rangle = 0$ and $\langle M \rangle = 1$). The reason behind this behaviour is that, for these values of magnetization, the configuration of the system is highly ordered, which makes temperature "kicks" of the system not to be able to break the symmetry of the system.

3.2 Dimer chain

For this geometry we used values for the exchange interaction $J_1 = 1$ (interaction between the two spins in one dimer) and $J_2 = 0.8$ (interaction between two spins not belonging to the same dimer). Inverse temperature was chosen to be $\beta = 100$. In this case we are interested in going to lower temperatures as the plateaus occur only at sufficiently low temperature. The results for magnetization curve and fluctuations are shown in figure 8.



(a) Magnetization as a function of the applied field. Inset in the top left shows the geometry studied.

(b) Temperature fluctuations of the magnetization (from figure 8a) as a function of the magnetization.

Figure 8: Magnetization curve and temperature fluctuations of the magnetization $\sigma(M)$ for a dimer chain of $N = 24$ spins with only nearest neighbour interactions at inverse temperature $\beta = 100$ averaged over 150 samples. The solid line shows the average and the shadowed region represents the fluctuations around the average values due to temperature fluctuations. Arrows show the spin orientation at highly ordered phases and boxes delimit the plateaus.

In this case, the magnetization curve shows a much more interesting behaviour than in the the 1D chain. The geometry of the system gives rise to magnetization plateaus at certain values of the magnetization. Here the quasi 2D geometry allows for spontaneous symmetry breaking giving rise to highly ordered phases [9]. One can interpret this situation as follows, the symmetry of the system opens an energy gap. Then the magnetic field applied to break this symmetry has to be large enough to overcome this gap, giving rise to flat regions in the magnetization curve.

It is important to remark that these plateaus are not perfectly flat due to the finite temperature of the system. The zero temperature magnetization curve would have shown perfectly flat plateaus [3]. However, temperature smooths out any sharp transition of the magnetization curve.

The first plateau is found at $\langle M \rangle = 0$ and corresponds to the two spins in one dimer to be anti-aligned in all of the dimers of the system. This plateau was not present in the 1D case even though an ordered phase was also found at zero magnetic field. This means that the presence of this plateaus is a spontaneous symmetry breaking phenomenon that comes from the geometry of the system.

The other three plateaus are found at magnetization values $\langle M \rangle = 1/4$, $\langle M \rangle = 1/3$ and $\langle M \rangle = 1/2$ which again are due to highly ordered and periodic phases of the system. In fact, it was proven [3] that at zero temperature, the dimer chain shows an infinite number of plateaus between $\langle M \rangle = 1/4$ and $\langle M \rangle = 1/2$ at magnetization values:

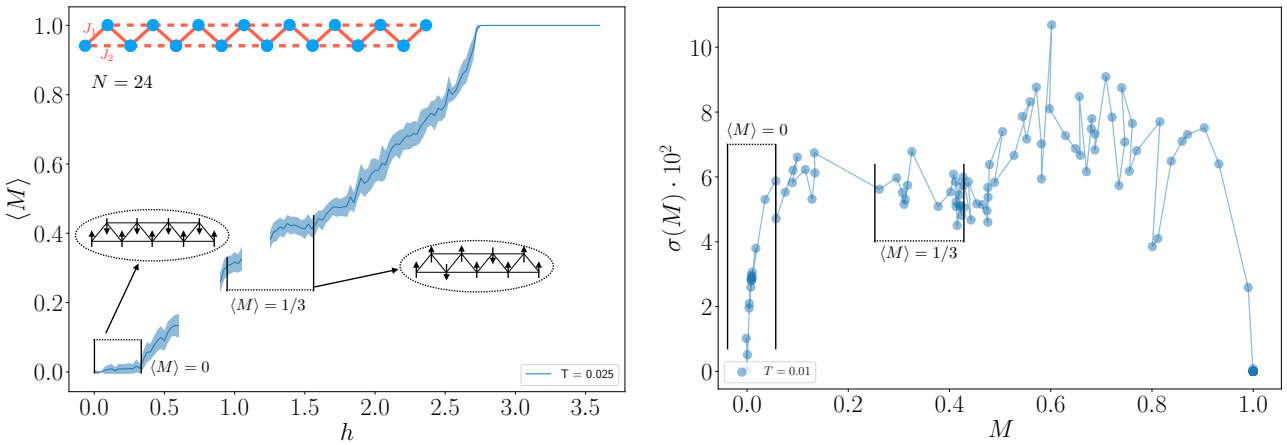
$$\langle M \rangle = \frac{k}{2k+2}, \quad k = 1, 2, 3, ... \tag{11}$$

This is because k consecutive vertical dimers form triplets ($S = 1$) and are separated by dimers forming singlets ($S = 0$).

Lastly, the effect of temperature and fluctuations in these ordered phases was analyzed in figure 8b. In this figure we can see that for $T = 0.001$ ($\beta = 100$) the values of magnetization corresponding to plateaus in the magnetization, show a significantly lower thermal fluctuations of the magnetization. Once again this is due to the fact that symmetry in highly ordered phases generates an energy gap bigger than the thermal energy of the system so temperature cannot take the system out of the ordered state. However for higher temperatures ($T = 0.02$ and $T = 0.025$ in figure 8b), the thermal energy of the system is bigger than this gap opened by symmetry so these ordered phases and plateaus are not observed.

3.3 1D next-nearest neighbour chain

The values used for the exchange constants were $J_1 = 1$ for the nearest neighbours interactions and $J_2 = 0.8$ for the next-nearest neighbours. Inverse temperature was chosen to be $\beta = 100$ as for the dimer case. The results for magnetization curve and fluctuations are shown in figure 9. Unfortunately, the magnetization curve is not complete as simulations are still running after nine days, However certain behaviours of the magnetization can already be identified.



(a) Magnetization as a function of the applied field. Inset in the top left shows the geometry studied.

(b) Temperature fluctuations of the magnetization (from figure 8a) as a function of the magnetization.

Figure 9: Magnetization curve and temperature fluctuations of the magnetization $\sigma(M)$ for a 1D next-nearest neighbour chain of $N = 24$ spins with only nearest neighbour interactions at inverse temperature $\beta = 100$ averaged over 50 samples. The solid line shows the average and the shadowed region represents the fluctuations around the average values due to temperature fluctuations. Arrows show the spin orientation at highly ordered phases and boxes delimit the plateaus. Gaps in the gaps are due to the lack of that data as the simulations did not finish in time

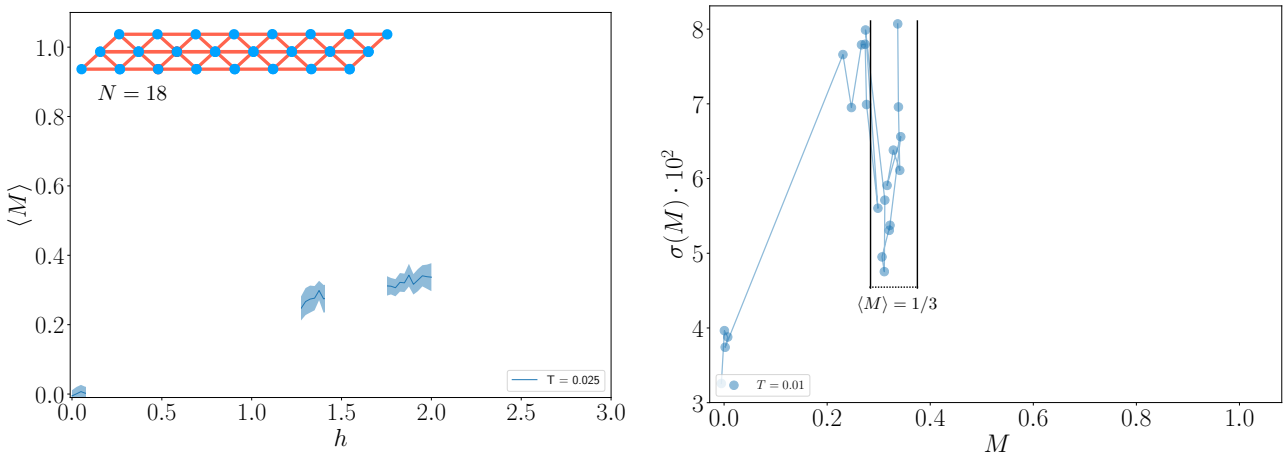
As in the dimer chain geometry, the 1D next-nearest neighbour chain shows magnetization plateaus, once again due to spontaneous symmetry breaking [9] and the opening of an energy gap. In this case only two highly ordered configurations were found. The first one corresponding with a Neel state with magnetization $\langle M \rangle = 0$. The last one, with $\langle M \rangle = 1/3$, is associated with an up-up-down configuration. The presence of these two configurations is a very interesting feature of this geometry. It is technically a 1D chain, so Neel states are expected at zero temperature and the longer-range interactions allow for the system to form a gap at zero magnetization. The up-up-down configuration is usually related to triangular geometries even though this is a 1D system. However, the next-nearest neighbour interaction term in the Hamiltonian makes the geometry of the system to effectively be a

triangular lattice (with $N_y = 2$) stretched along the direction of one of its primitive axis's. This is the reason why this system shows ordering typical of both a linear and triangular geometry.

Finally, it is important to remark that once again the thermal fluctuations of the magnetization provide a good criteria to determine if there is a plateau at a certain magnetization values as as the fluctuations became smaller in the plateau phases as shown in figure 9b.

3.4 Triangular lattice

Unfortunately, the complete magnetization curve did not finish running on time. The simulations are still going after nine days. However the results obtained so far are shown in figure 10. The triangular lattice, as mentioned before, has a geometry in which every site has six nearest neighbours, making the bonds dimensions of the MPS to be significantly larger than the ones found in the easier geometries. Future work will be to analyze the results coming from this simulations. Even though the results shown are very preliminary and incomplete, they are promising if one pays attention to the thermal fluctuations in figure 10b, as they seem to drop at $\langle M \rangle = 1/3$.



(a) Magnetization as a function of the applied field. Inset in the top left shows the geometry studied.

(b) Temperature fluctuations of the magnetization (from figure 10a) as a function of the magnetization.

Figure 10: Magnetization curve and temperature fluctuations of the magnetization $\sigma(M)$ for a 1D next-nearest neighbour chain of $N = 18$ spins with only nearest neighbour interactions at inverse temperature $\beta = 100$ averaged over 50 samples. The solid line shows the average and the shadowed region represents the fluctuations around the average values due to temperature fluctuations. Arrows show the spin orientation at highly ordered phases and boxes delimit the plateaus. Gaps in the gaps are due to the lack of that data as the simulations did not finish in time

4 Conclusion and future work

METTS algorithm successfully reproduced the behaviour of the experimental results. Magnetization plateaus were found at finite temperature for different geometries. As expected, no plateaus were found for the 1D chain with nearest neighbour interactions, whereas geometries with longer range interactions showed these plateaus for sufficiently low temperatures, as observed experimentally.

METTS algorithm is a very powerful technique to study quantum many body problems as the computational resources required to implement it do not scale exponentially with the size of the system, which is one of the biggest issues when studying the quantum many body problem. In this case, systems with $N = 24$ and $N = 27$ spins were successfully studied, which would have been intractable if exact diagonalization was chosen to solve the problem as the dimension of the Hilbert spaces required to represent the states are $17 \cdot 10^6$ and $132 \cdot 10^6$ respectively.

Furthermore, the number of samples required to achieve statistically accurate results is also fairly small (no larger than hundreds) compared to the number of samples required to achieve the same

accuracy using Monte Carlo techniques, that usually requires the collection of thousands of samples.

The only drawback of this algorithm is that it becomes slow when going to low temperatures. Due to the short time to complete the project I chose a very low temperature to run the simulations to be sure I would find the plateaus, which made the run-time of the simulations to be of a few days. In the future I want to explore what is the critical temperature for which the plateau phase disappears for different geometries. This temperature dependant study will also help understanding the effect of temperature on these phases. The ultimate goal would be to construct a phase diagram with the obtained results and match the computed magnetization curves with experimental results.

5 Implementation

The intent of this section is to describe the technical detains on how the algorithm was implemented, as well as describing the codes submitted with this report. The programming language chosen to do the calculations and obtain the data was C++ while the data analysis and representation was done in Python. C++ was chosen because of its higher efficiency and faster run-times, but more importantly, because the library "ITensor" was used to implement the algorithm [10] and this library is implemented in C++. This library was used as the algorithm requires the heavy use of tensors as described above as well as basic tools of linear algebra implemented in "ITensor", such as SVD's or the notion of moving orthogonality centres. "ITensor" is a similar version of "numpy" but to work with tensors. It allows to define objects such as indexes, tensors, MPS's.... This library can be downloaded from their website [10] or github and then installed and compiled. Detailed instructions on how to install it can be found in the library website.

In order to run my implementation of the code, four files are needed:

- "Makefile". This is the file that has all the information to compile the code. Running "make" in the terminal will access this file and compile the code with the options specified in this file. Detailed instructions on what has to be included in this file are in the file itself. But generally, the script with the code driving the simulations has to be specified. Running "make" in the terminal will compile the code and create an executable (named "METTS" in this case). Then in order to run the program, one has to run "./METTS inputfile_metts". This brings us to the next item in the list:

- "inputfile_metts". This file contains all the parameters that the program will use to run: "N" is the number of spins in the system, "hsteps" is the number of magnetic field values to collect for the magnetization curve,"hinit" is the first value of magnetic field and "hstep" is the increments in magnetic field, "J1" and "J2" are the exchange interaction constants between first and second neighbours in the lattice respectively, "beta" is the inverse temperature, "tau" is the τ value for the Trotter decomposition, "cutoff" is a variable such that if in a SVD we find a singular value smaller than "cutoff" then we set it to zero, "maxm" is the maximum bond dimenssion allowed in a MPS, "Malias" is the number of METTS genrated before starting to take samples to compute physical quantities and "MMETTS" is the number of samples used to average over physical quantities.

- "METTS.cc". This is the code that drives the simulation. It imports all the parameters for the simulation, and allows to define the physical system that you want to study (from the ones described in this report). Then it calls the METTS algorithm which is implemented in "tools.h". Then it saves the results of the magnetization returned by METTS in a text file called "mH.txt".

- "tools.h". This is the code will all the functions required to run the algorithm. It includes functions: "collapse" that performs the collapse step of the algorithm, "measure" that measures the magnetization as explained above, "HAMILTONIAN" that creates and exponentiates a Heisenberg Hamiltonian for the geometry specified by the user between the geometries described in this report, "METTS_MAG" that calling the functions above implements the algorithm and returns an array with the magnetization values for each METTS generated.

References

[1] E. M. Stoudenmire and Steven R. White, New Journal of Physics **12**, 055026 (2010).

[2] G. Allodi, R. De Renzi, S. Agrestini, C. Mazzoli, and M. R. Lees, Physical Review B **83**, 104408 (2011).

[3] A Honecker et al, J. Phys.: Condens. Matter **16**, S749 (2004).

[4] Yong Hu, J. Phys.: Condens. Matter **20**, 125225 (2008).

[5] M. E. Zhitomirsky, A. Honecker, and O. A. Petrenko, Physical Review Letters **15**, 3269-3272.

[6] Zi-Xiang Li, Hong Yao, arXiv:1805.08219v2.

[7] Nielsen and Chuang, "Quantum Computation and Information", 10th anniversary edition, Cambridge University Press.

[8] William H. Press, Saul A. Teukolsky, William T. Vetterling and Brian P. Flannery, "Numerical recipes", third edition, Cambridge University Press.

[9] Bertrand I. Halperin, arXiv:1812.00220.

[10] http://itensor.org/index.html