

# WORLD-LINE QUANTUM MONTE CARLO



December 2025



Victor Caro & Théa Araujo



## Abstract

The objective of this project is to implement the algorithm proposed by Assaad and Evertz [1]: a world-line approach to quantum Monte Carlo methods for one-dimensional XXZ Heisenberg spin chains.

## CONTENTS

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	The XXZ Hamiltonian . . . . .	2
1.2	Computational complexity . . . . .	3
<b>2</b>	<b>World-line approach</b>	<b>3</b>
2.1	Trotterization . . . . .	3
2.2	Weight and energy of world-lines . . . . .	3
2.3	Generating world-line configurations . . . . .	5
<b>3</b>	<b>Metropolis algorithm</b>	<b>5</b>
3.1	Local shifts . . . . .	6
3.2	Vertex update . . . . .	6
3.3	Loop updates . . . . .	7
<b>4</b>	<b>Discussion</b>	<b>10</b>
4.1	Energy . . . . .	10
4.2	Spin correlation . . . . .	11
4.3	Efficiency of the different update methods . . . . .	13

## 1 INTRODUCTION

Problems in solid-state physics often require computationally demanding calculations, as their complexity typically scales as a power of system size. Spin chains and electron chains exhibit strong correlations, making exact solutions infeasible for large systems. However, one-dimensional (1D) spin chains provide a good balance between analytical tractability and physical relevance. In this study, the focus is placed on 1D XXZ chains, where correlations along the  $x$  and  $y$  axes are identical. This framework allows verification of theoretical predictions, assessment of approximations, and extrapolation of system properties to larger sizes. The Monte Carlo method described in [1] was implemented using three distinct update schemes.

### 1.1 The XXZ Hamiltonian

The XXZ model describes a chain of interacting spin- $\frac{1}{2}$  particles with nearest-neighbour coupling and anisotropy between the  $xy$ -plane and the  $z$ -direction. The Hamiltonian is given by

$$H = J_x \sum_i (S_i^x S_{i+1}^x + S_i^y S_{i+1}^y) + J_z \sum_i S_i^z S_{i+1}^z \quad (1)$$

Periodic boundary conditions are imposed such that  $S_i = S_{i+n}$ . The coupling constants  $J_x$  and  $J_z$  define the physical regime of the system: the  $J_z$  term represents an Ising-like interaction, favouring spin alignment ( $J_z < 0$ ) or anti-alignment ( $J_z > 0$ ), whereas  $J_x$  induces quantum fluctuations through spin exchanges between neighbouring sites.

## 1.2 Computational complexity

Although the Hamiltonian can be written compactly, evaluating thermodynamic quantities such as specific heat, magnetic susceptibility, or spin stiffness requires computation of the partition function  $Z = \text{Tr}[e^{-\beta H}]$ .

This presents a mathematical challenge: the Hilbert space of a quantum system grows exponentially with its physical size. Since each local spin has two basis states ( $|\uparrow\rangle$  and  $|\downarrow\rangle$ ), the dimension of the total Hilbert space is:

$$\dim(\mathcal{H}_n) = 2^n \quad (2)$$

For very small systems, one can perform diagonalization of the Hamiltonian matrix. However, this approach becomes impossible for large systems. Therefore, it is necessary to use other numerical approaches to solve for larger systems that do not require the full matrix diagonalization. Nevertheless, this exact calculation is crucial for testing the numerical method on smaller systems and for extrapolating to larger ones.

## 2 WORLD-LINE APPROACH

### 2.1 Trotterization

Following the derivation by Assaad and Evertz [1], the problem is simplified to make it tractable numerically. Let  $H^{(i)} = J_x(S_i^x S_{i+1}^x + S_i^y S_{i+1}^y) + J_z S_i^z S_{i+1}^z$  be the two-site Hamiltonian, whose eigenvalues are known. Then,  $H_1$ , the sum of the even two-sites, and  $H_2$ , the sum of the odd terms, are two sums of commuting Hamiltonians. Introducing the discretisation  $\Delta\tau \equiv \frac{\beta}{m}$ , Trotter's formula approximates the partition function as:

$$\begin{aligned} Z &= \text{Tr}[(e^{-\Delta\tau H})^m] \\ &= \text{Tr}[(e^{-\Delta\tau H_1} e^{-\Delta\tau H_2})^m] + \mathcal{O}(\Delta\tau^2) \\ &= \sum_{\sigma_1 \dots \sigma_{2m}} \langle \sigma_1 | e^{-\Delta\tau H_1} | \sigma_{2m} \rangle \dots \langle \sigma_2 | e^{-\Delta\tau H_2} | \sigma_1 \rangle + \mathcal{O}(\Delta\tau^2) \end{aligned}$$

Then,  $\Delta\tau$  can then be interpreted as a discrete imaginary time step, mapping the 1D system into a 2D configuration with a progression from  $\tau = 0$  to  $\tau = m\Delta\tau = \beta$  as the additional dimension. This conversion allows to treat this system with a classical approach, thus enabling Monte Carlo methods.

This system can then be represented as a world-line configuration tracking the evolution of the chain's spin configurations through time and space. This representation possesses interesting properties, such as spin conservation, which simplifies the calculation by eliminating spin evolution or 2D configurations not having a meaningful contribution at first order on  $\Delta\tau$ .

Periodic boundary conditions are chosen as they ensure that boundary effects are minimised. In practice,  $m$  needs to be large enough so as to minimise the effects of the Trotterization.

As for the code, a **Worldline** is represented by a matrix of size  $2m \times n$  containing  $\pm 1$  values depending on the state of the spin at that location.

### 2.2 Weight and energy of world-lines

Each **Worldline** has its own weight denoted as  $\Omega(\omega)$ .

From [1]:

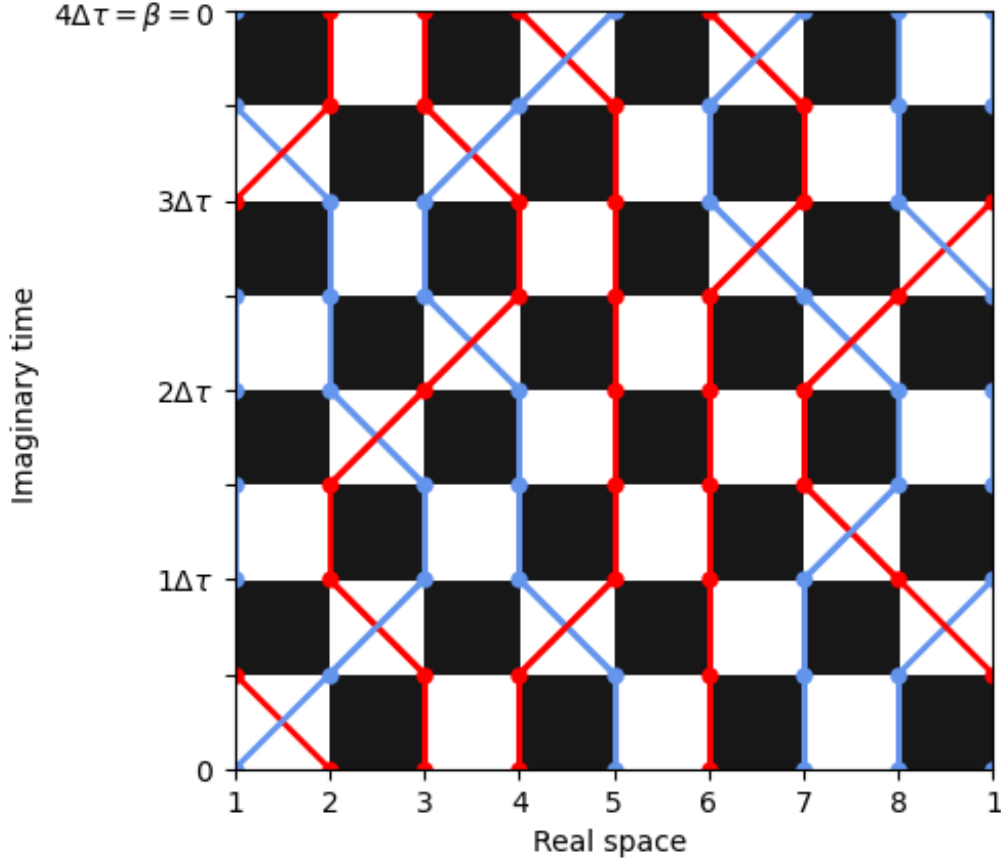


Figure 1: World-line configuration for 8 spin sites and  $m = 4$ . Red represents up spins and blue represents down spins.

$$\Omega(\omega) = \prod_{i=1}^{2m} \langle \sigma_{i+1} | e^{-\Delta\tau H_i} | \sigma_i \rangle \quad (3)$$

Where  $H_i$  is  $H_1$  if  $i$  is odd and  $H_2$  otherwise. Periodic boundary conditions give  $\sigma_{2m+1} = \sigma_1$ .

With eq. (3), the total weight is the product of the weight of each line.

$$\langle \sigma_{i+1} | e^{-\Delta\tau H_i} | \sigma_i \rangle = \prod_{j=i}^{n/2} \langle \sigma_{2j,i+1}, \sigma_{2j+1,i+1} | e^{-\Delta\tau H_i} | \sigma_{2j,i}, \sigma_{2j+1,i} \rangle \quad (4)$$

And eq. (4) gives that each line weight is the product of the weight of each of its white plaquette.

*Note: In the code  $i$  and  $j$  are switched from the usual convention to match the notation used in the paper.*

The combination of eq. (3) and eq. (4) gives a straightforward algorithm to compute the weight of a Worldline:

$$\Omega(\omega) = \prod_p W_p \quad (5)$$

Where  $p$  is taken over all white plaquettes of  $\omega$  and  $W_p$  is the plaquette weight.

The energy associated with a configuration is: (from [1])

$$E(w) = -\frac{1}{m} \sum_p \frac{\partial}{\partial \Delta\tau} \ln(W_p) \quad (6)$$

Based on the weights provided in [1] :

- **Full Squares:**

$$W = e^{-\Delta\tau J_z/4} \quad (7)$$

$$-\frac{\partial}{\partial \Delta\tau} \ln(W) = -\frac{\partial}{\partial \Delta\tau} \left( -\frac{\Delta\tau J_z}{4} \right) = \frac{J_z}{4} \quad (8)$$

- **Vertical World-Lines:**

$$W = e^{\Delta\tau J_z/4} \cosh \left( \frac{\Delta\tau J_x}{2} \right) \quad (9)$$

$$-\frac{\partial}{\partial \Delta\tau} \ln(W) = -\left( \frac{J_z}{4} + \frac{J_x}{2} \tanh \left( \frac{\Delta\tau J_x}{2} \right) \right) \quad (10)$$

- **Crossed World-Lines:**

$$W = -e^{\Delta\tau J_z/4} \sinh \left( \frac{\Delta\tau J_x}{2} \right) \quad (11)$$

$$-\frac{\partial}{\partial \Delta\tau} \ln |W| = -\left( \frac{J_z}{4} + \frac{J_x}{2} \coth \left( \frac{\Delta\tau J_x}{2} \right) \right) \quad (12)$$

### 2.3 Generating world-line configurations

In order to test the validity of Trotter's formula and our calculation method, one can generate all possible valid configurations for small grids. This allows to make exact calculations and compare them to our results.

The first approach took was to generate all the spin configurations possible and to keep only the valid ones. While straightforward to implement, this approach was computationally inefficient, as it required computing  $2^{2mn}$  configurations.

The second approach consisted of recursively generating a world-line with backtracking whenever an invalid configuration was encountered. Enforcing periodic boundary conditions proved more difficult than expected, resulting in reduced implementation efficiency. Each world-line was generated until an invalid configuration appeared on a white plaquette, at which point the process was halted. However, this algorithm did not yield significant improvement over the previous one, as the number of configurations continued to grow exponentially. The method could be optimised by detecting invalid configurations earlier to avoid generating multiple invalid branches. Using this approach, all valid configurations were generated for systems with fewer than six sites and  $m \leq 4$  in under one minute. For eight sites at  $m = 4$ , computational limitations prevented full enumeration, although the number of valid configurations is estimated to exceed  $10^6$ .

The generation of all world-lines enabled the computation of the Trotterization error which is shown in fig. 6.

## 3 METROPOLIS ALGORITHM

To perform a complete Monte Carlo simulation, the Metropolis algorithm [2] was used and three different update schemes were implemented.

n/m	1	2	3	4
2	6	18	66	258
4	18	90	546	3618
6	66	546	6840	94386
8	258	3618	94386	-

Table 1: Number of valid configurations for n sites and different values of m

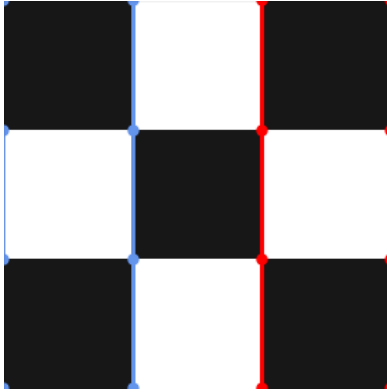
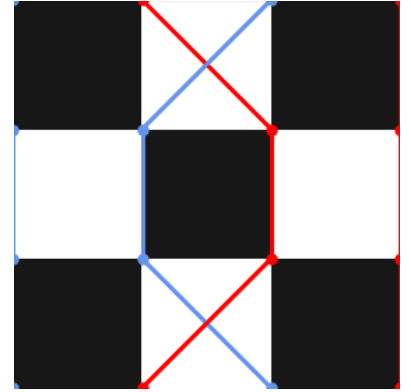
(a) Configuration *side*(b) Configuration *crossed*

Figure 2: A local shift move

### 3.1 Local shifts

The first update approach is by shifting spin lines on a shaded plaquette. An illustration of this can be seen in fig. 2.

The advantage of this technique is its ease of implementation as the energy and weight shifts are trivial to compute, and the move has a symmetric probability.

The major drawback of this update method is that it is not ergodic. Indeed, the number of up spins remains the same throughout the move. This restricts the simulation to a subspace of the problem.

### 3.2 Vertex update

The second update method is inspired by [1] and uses the mapping to the 6-vertex problem.

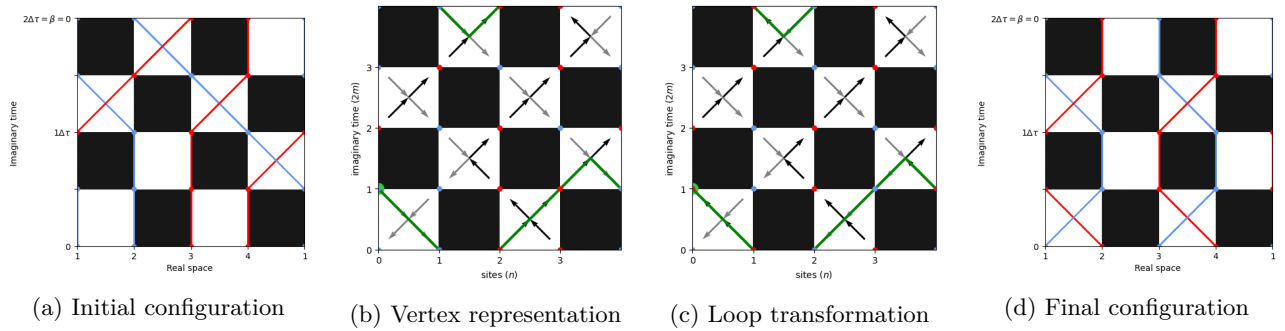


Figure 3: Metropolis algorithm with vertex update

Once the problem is mapped to the vertex representation, a random location is selected, and a random path is traced following the vertex orientations. This path necessarily returns to its starting point, forming a closed loop. The directions of the vertices encountered along the path are then inverted, producing another valid world-line configuration. Prior to the flip, the weight associated with the exchange along the loop is computed, as the move is subject to the Metropolis acceptance criterion. The transition probability is symmetric.

Any two configurations can be connected by flipping an appropriate set of loops, demonstrating ergodicity.

**The configurations differ by a finite number of loops.** – Consider two configurations,  $A$  and  $B$ , in their six-vertex representations. For each white plaquette, either  $A$  and  $B$  share the same vertex configuration or they contain two arrows in opposite directions. In the latter case, exactly two arrows must be reversed, since each square possesses two outgoing and two incoming arrows. Because the arrows form continuous loops across all world-lines, the set of differing arrows between  $A$  and  $B$  corresponds to a collection of closed loops.

**All loops are possible** – Since paths are generated randomly and the visited plaquettes are chosen at random, any loop present in the configuration space can, in principle, be produced by the algorithm.

### 3.3 Loop updates

The last update method implemented is the one described in [1].

It is first observed that each vertex configuration can be traversed in three distinct ways. Consequently, each plaquette is associated with three possible graphs, as illustrated in fig. 4.

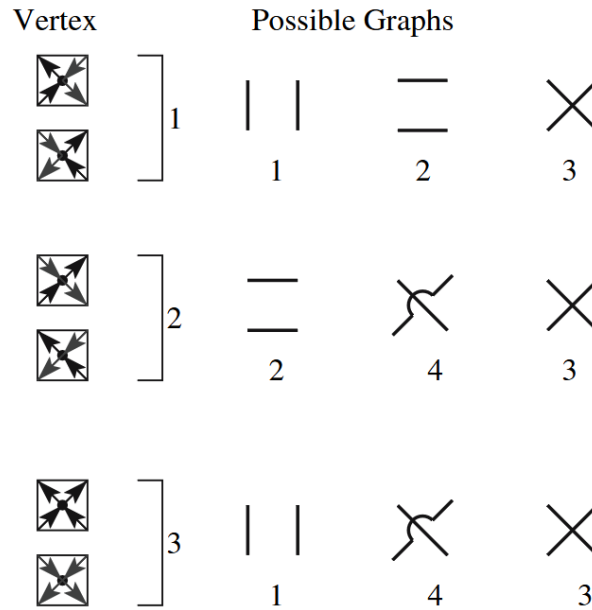


Figure 4: Correspondence between vertex drawings and graphs. Taken from Assaad and Evertz [1] and adapted.

Weights are defined for each plaquette–graph combination as  $W(S, G)$ , where  $S \in \{1, 2, 3\}$  denotes the plaquette type and  $G \in \{1, 2, 3, 4\}$  denotes the graph, such that the normalisation condition holds.

$$W(S) = \sum_G W(S, G) \quad (13)$$

From fig. 4, it follows that  $W(1, 4) = W(2, 1) = W(3, 2) = 0$ .

These weights are used to define the probability associated with selecting a particular graph.

$$\mathbb{P}(S \rightarrow (S, G)) = \frac{W(S, G)}{W(S)} \quad (14)$$

Once each plaquette has been mapped to a corresponding graph, multiple loops spanning the entire world-line configuration are obtained. A Union-Find data structure is employed to construct these loops, as illustrated in fig. 5.

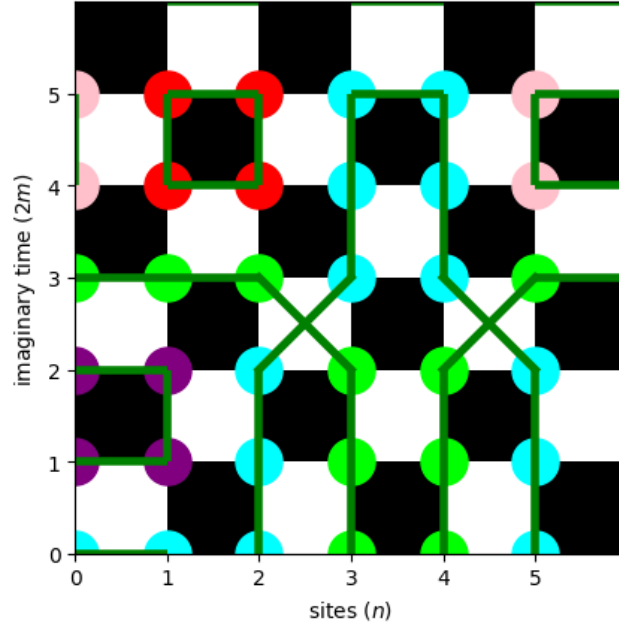


Figure 5: Partition of the word-line in loops. The green lines are the graphs. *Note: the crossed lines represents graph 4 and not 3.*

For each loop defined by the graphs, it is then determined whether to flip the spins. The probability of such a flip is computed accordingly:

$$\mathbb{P}((S, G) \rightarrow (S', G)) = \frac{W(S', G)}{W(S, G) + W(S', G)} \quad (15)$$

To ensure symmetric transition probabilities in the Metropolis algorithm, it is required that for any  $G$  and compatible  $S$  and  $S'$ , the symmetry condition holds:

$$W(S, G) = W(S', G) \quad (16)$$

By combining eq. (15) and eq. (16), the following relation is obtained.

$$\mathbb{P}((S, G) \rightarrow (S', G)) = \frac{1}{2} \quad (17)$$



Moreover the detailed balance is given by:

$$\begin{aligned}
W(S)\mathbb{P}(S \rightarrow S') &= W(S) \sum_G \mathbb{P}(S \rightarrow (S, G)) \mathbb{P}((S, G) \rightarrow (S', G)) \\
&= \sum_G W(S) \frac{W(S, G)}{W(S)} \mathbb{P}((S, G) \rightarrow (S', G)) \\
&= \sum_G W(S', G) \mathbb{P}((S, G) \rightarrow (S', G)) \\
&= \sum_G \frac{W(S')}{W(S')} W(S', G) \mathbb{P}((S', G) \rightarrow (S, G)) \\
&= \sum_G W(S') \frac{W(S', G)}{W(S')} \mathbb{P}((S', G) \rightarrow (S, G)) \\
&= W(S') \sum_G \mathbb{P}(S' \rightarrow (S', G)) \mathbb{P}((S', G) \rightarrow (S, G)) \\
&= W(S') \mathbb{P}(S' \rightarrow S)
\end{aligned}$$

Thus, each loop has a probability of one half to be flipped.

The expressions for  $W(S, G)$  are then derived. Following eq. (13) and eq. (16), the system of equations can be written as:

$$\begin{aligned}
W(1) &= W(1, 1) + W(1, 2) + W(1, 3) \\
W(2) &= W(1, 2) + W(2, 4) + W(1, 3) \\
W(3) &= W(1, 1) + W(2, 4) + W(1, 3)
\end{aligned}$$

The system contains four variables but only three independent equations. The parameter  $W(1, 3) = f$  is selected as a free variable. After algebraic manipulation, the following expressions are obtained:

$$\begin{aligned}
W(1, 1) &= \frac{1}{2} (W(1) + W(3) - W(2) - f) \\
W(1, 2) &= \frac{1}{2} (W(1) + W(2) - W(3) - f) \\
W(2, 4) &= \frac{1}{2} (W(2) + W(3) - W(1) - f)
\end{aligned}$$

To ensure that all graph weights remain positive, the resulting conditions on the weight values must be satisfied:

$$\begin{aligned}
W(1) &\leq W(2) + W(3) \\
W(2) &\leq W(1) + W(3) \\
W(3) &\leq W(1) + W(2)
\end{aligned}$$

For implementation simplicity, the parameter  $f$  is set to zero, which restricts each plaquette to only two possible graphs.

The ergodicity of the method follows from the previous demonstration of ergodicity. Since any two configurations differ by a finite set of loops, and the present algorithm can generate and flip such loops, the method is therefore ergodic.

## 4 DISCUSSION

Different update methods were compared by computing physical observables for small system sizes, where exact results are available for validation. These benchmarks were then extended to larger systems.

All Monte Carlo simulations were performed with identical parameters:  $n_{\text{cycle}} = 5000$ ,  $\text{length\_cycle} = 2mn$ ,  $n_{\text{rep}} = 10$ . Error bars represent the standard deviation across the  $n_{\text{rep}}$  runs scaled down by  $\sqrt{n_{\text{rep}}}$ .

The formulation from Assaad and Evertz [1] was used to correct for Trotterization errors and compute observables:

$$\langle O \rangle = \frac{\text{Tr} [e^{-\beta H} O]}{\text{Tr} [e^{-\beta H}]} = \frac{\sum_w \Omega(w) O(w)}{\sum_w \Omega(w)} \quad (18)$$

The largest system for which exact results were available was an 8-spin chain, simulated for several values of  $m$ .

### 4.1 Energy

The energy was computed for the three update schemes. As discussed in Section 3.1, the local shift method is not ergodic and therefore conserves the total spin, restricting the accessible configuration space.

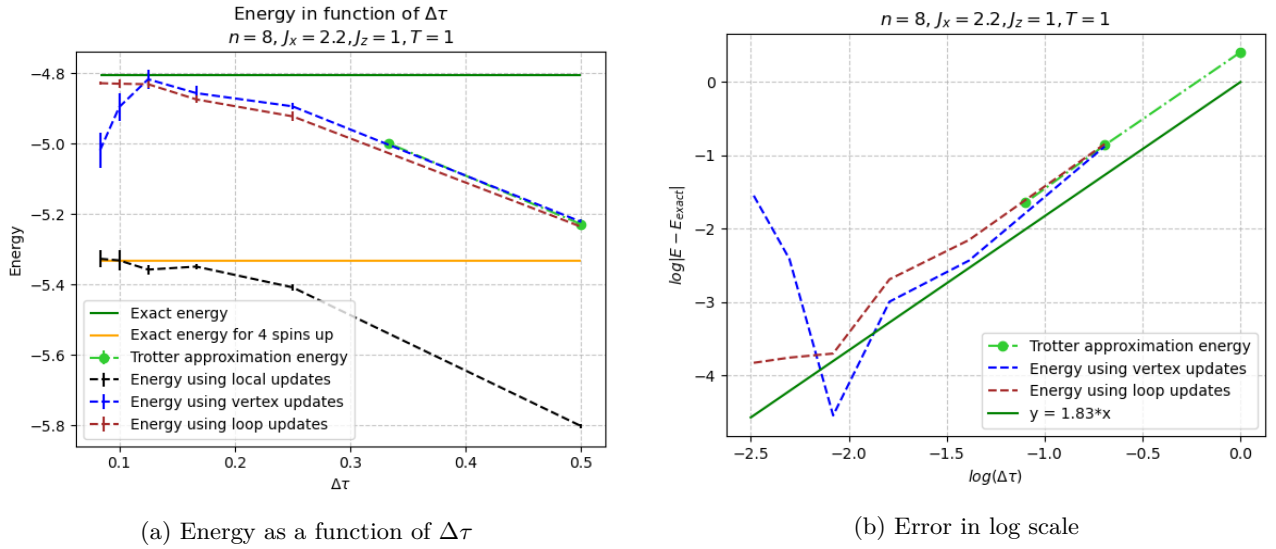
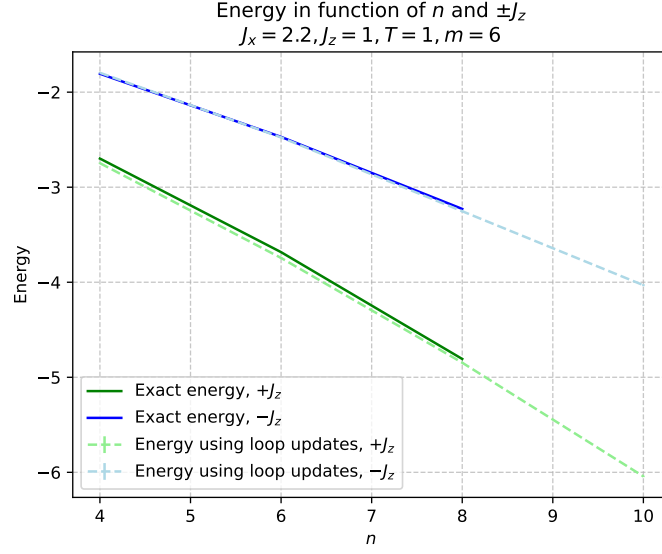


Figure 6: Energy for 8 spin sites and different values of  $m$  plotted for different update methods

The energy error for all methods is dominated by the Trotter approximation at large  $\Delta\tau$ . However, for smaller values of  $\Delta\tau$ , the computed energies agree closely with the exact values, validating the model and suggesting the criterion  $\Delta\tau \sim 0.1J^{-1}$  for accurate calculations. fig. 6b indicates that the error scales as  $\mathcal{O}(\Delta\tau^{1.83})$ , which is consistent with the expected  $\mathcal{O}(\Delta\tau^2)$ . The deviation observed for the vertex update is discussed in section 4.3.

The advantage of the numerical methods becomes clear for larger systems, such as 10 spin sites and  $m = 6$  as can be seen in fig. 7. The loop updates algorithm is able to calculate the energy for this larger system whereas the exact calculation was not possible.

Figure 7: Energy as a function of the number of sites for  $m = 6$ 

## 4.2 Spin correlation

The model also allows computation of spin–spin correlation functions. Since each line represents a valid spin configuration, the correlation operator was averaged over all positions and imaginary-time slices:

$$\langle S^z S_d^z \rangle = \frac{1}{2mn} \sum_{j,i} S_{j,i} S_{j,i+d} \quad (19)$$

The exact correlation is computed as in eq. (20) only on the first site (the value does not depend on the site).

$$\langle S^z S_d^z \rangle = \frac{\text{Tr}(e^{-\beta H} S_0^z S_d^z)}{\text{Tr}(e^{-\beta H})} \quad (20)$$

The simulation accurately reproduced the correlations in both the ferromagnetic and antiferromagnetic regimes (fig. 8).

*Note: Other update schemes were not employed for this observable due to insufficient accuracy in their energy estimates.*

The influence of the coupling  $J_x$  on spin correlations was also examined (fig. 9). Increasing  $J_x$  strengthens spin correlations, as reflected by the reduction of green regions in the heat map.

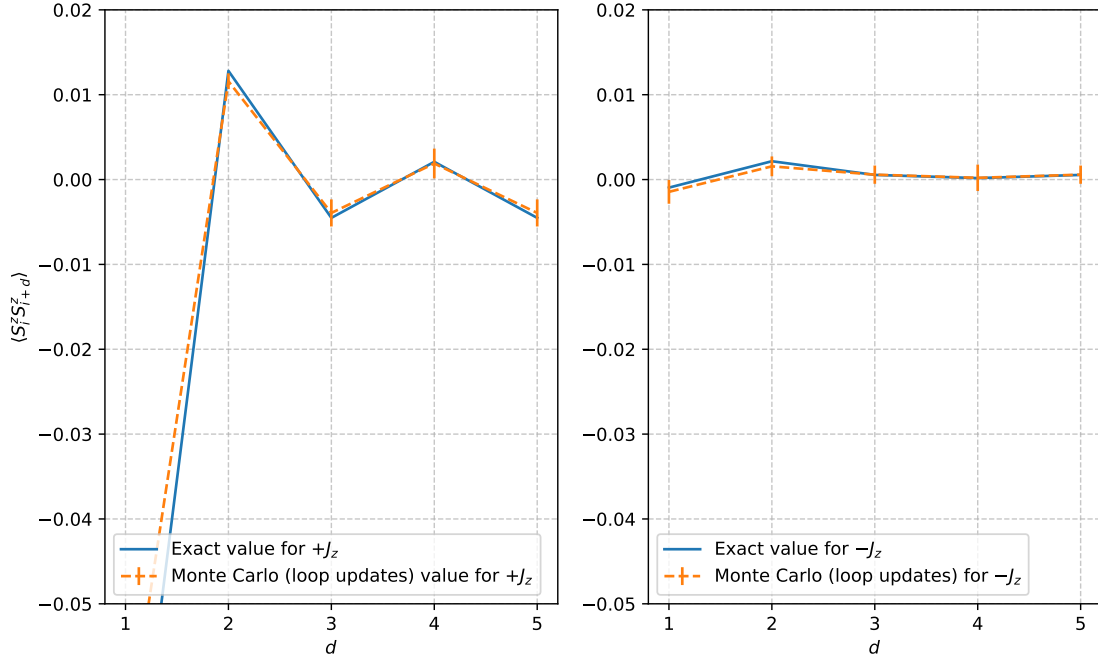
Spin correlation for  $n = 8, J_x = 2.2, J_z = \pm 1, T = 1, m = 6$ 

Figure 8: Spin correlation computed using loop updates

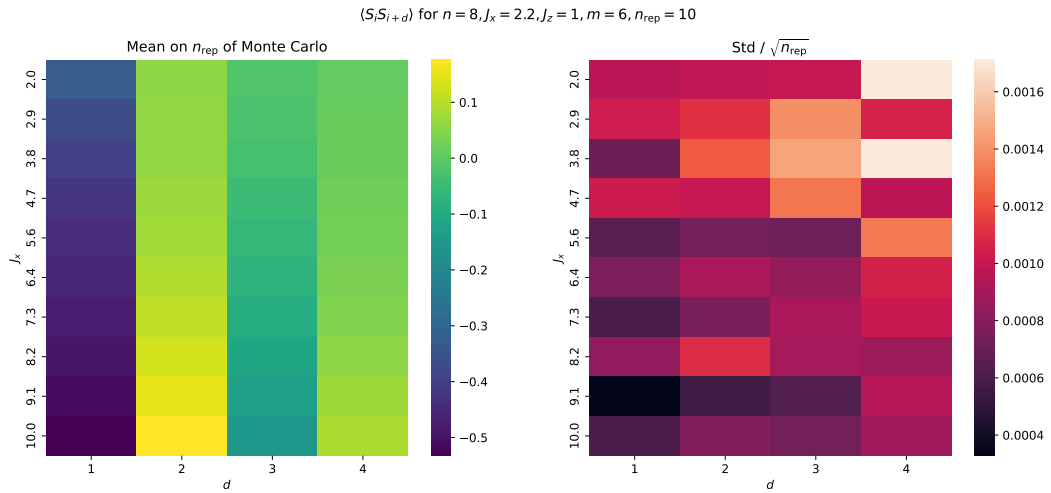


Figure 9: Spin correlation as a function of  $J_x$ . The graphs must be understood as follows : the right heat-map shows mean correlations values on the  $n_{\text{rep}}$  runs and the left heat-map gives the uncertainty ( $\text{std}/\sqrt{n_{\text{rep}}}$ ) on each of the mean values of the right graph.

### 4.3 Efficiency of the different update methods

The efficiency of each scheme was evaluated using two metrics: computation time (fig. 10) and acceptance rate (table 2).

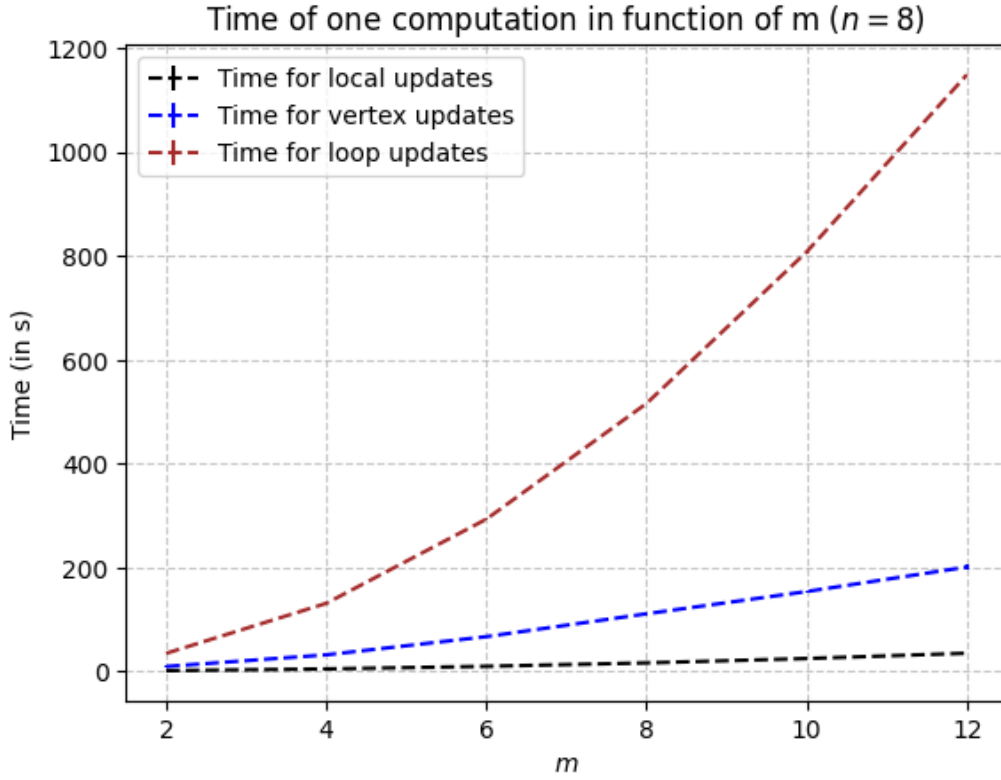


Figure 10: Computation time for different update algorithms

m	2	4	6	8	10	12
Acceptance rate	46,6 %	13,2 %	5,9 %	4,0 %	3,3 %	2,5 %

Table 2: Acceptance rate for vertex update for  $10^5$  tries

Although the loop update requires longer computation times, its acceptance rate remains close to 100%, meaning that almost every proposed update results in a configuration change. However, for the vertex update the acceptance rate drops significantly as  $m$  increases. Thus, for large  $m$ , roughly  $50\times$  more trials are required to accept an update. This can be seen in fig. 6: for smaller  $m$ , the accuracy of the vertex update decreases. This is probably due to an insufficient amount of accepted updates leading to inaccurate calculations. To conclude, the high acceptance rate of the loop update is essential for obtaining accurate results on large systems, justifying its longer computation time. This qualitative analysis could be verified by calculating the autocorrelation time which measures the necessary number of steps to obtain a decorrelated configuration. Efficiency could be defined as the inverse of the product of the autocorrelation and computation times.

In summary, the loop update proved to be the most accurate and efficient method implemented.

## REFERENCES

- [1] F.F. Assaad and H.G. Evertz. “World-line and Determinantal Quantum Monte Carlo Methods for Spins, Phonons and Electrons”. In: *Computational Many-Particle Physics*. Ed. by H. Fehske, R. Schneider, and A. Weiße. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 277–356. ISBN: 978-3-540-74686-7. DOI: 10.1007/978-3-540-74686-7\_10. URL: [https://doi.org/10.1007/978-3-540-74686-7\\_10](https://doi.org/10.1007/978-3-540-74686-7_10).
- [2] Nicholas Metropolis and Stanislaw Ulam. “The monte carlo method”. In: *Journal of the American statistical association* 44.247 (1949), pp. 335–341.