3 Algorithm

In this section, we describe the algorithm starting from the relatively general concepts and ideas exposed in G. Rakala and K. Damle's paper [15] and building towards more concrete details of our implementation.

3.1 An introduction to the Metropolis-Hastings algorithm

To facilitate the discussion of the algorithm, let us just take some time to quickly review the basics of the Metropolis-Hastings algorithm. All of the following can be found in much more details on Wikipedia or in textbooks such as the lecture notes by A. W. Sandvik [20] or the book by M. Benaïm and N. El Karoui [21].

3.1.1 Markov chains

The Metropolis-Hastings algorithm is an example of a Markov Chain Monte Carlo (or MCMC) method, so let us have a quick look at the definition and some properties of Markov chains.

Definition of a Markov chain A *stochastic process* is the evolution of a random variable. The evolution can have either discrete or continuous time. Such a process is called a *Markov process* (named after Andreï Markov) provided it satisfies the *Markov property*, that is, the relevant information allowing the prediction of the future of the random variable is fully contained in the current state of the process. This means that taking into account information from previous steps won't give any further insight; the process is said to be *memoryless*. The definition of a *Markov chain* can vary, but, as far as we are concerned, we can consider that a Markov chain is a Markov process with discrete time and discrete state space E. So, to summarize, we are considering a set of random variables X_n on the same discrete probability space and indexed by the discrete time n. They have the Markov property, namely:

$$\mathcal{P}(X_{n+1} = j | X_n = i_n, ..., X_0 = i_0) = \mathcal{P}(X_{n+1} = j | X_n = i_n) \tag{11}$$

Transition probabilities and homogeneous Markov chain The *transition probability from state i to state j in one step* is defined as:

$$p_{ij} := \mathcal{P}(X_1 = j | X_0 = i) \tag{12}$$

The Markov chain is said to be *time homogeneous* if

$$\mathcal{P}(X_{n+1} = j | X_n = i) = \mathcal{P}(X_n = j | X_{n-1} = i) \tag{13}$$

From now on we will be considering time homogeneous Markov chains. Thus the transition probability p_{ij} doesn't depend on time. The *transition matrix* of the Markov chain is defined as $P = (p_{ij})_{(i,j)\in E^2}$. The law of the Markov chain is characterized by its initial distribution $\mathcal{P}(X_0 = i_0)$ and its transition matrix. Indeed,

$$\mathcal{P}(X_0 = i_0, X_1 = i_1, ..., X_n = i_n) = \mathcal{P}(X_0 = i_0) p_{i_0 i_1} p_{i_1 i_2} ... p_{i_{n-1} i_n} \tag{14}$$

Stationary distribution If E is a subspace of states and P is the transition matrix of a Markov chain on this subspace, then we call $\pi = (\pi_i)_{i\in E}$ a *stationary measure over* E if it is an eigenfunction of P of eigenvalue 1. If, moreover, $\pi_i \geq 0\ \forall i \in E$ and $\sum_{i\in E} \pi_i = 1$, then π is a *stationary distribution* of the Markov chain. We will neither explore the detailed requirements to ensure the existence and unicity of the stationary distribution here, nor will we detail the conditions of convergence to the stationary distribution. We will notice however that if the law of X_n is a stationary distribution π, then for all $k \geq n$, the law of X_k is still π.

3.1.2 Markov Chain Monte Carlo methods

Monte Carlo methods are algorithmic methods aiming at computing a numerical estimate using stochastic processes. The purpose is to generate randomly N states $(x_1, x_2, ..., x_N)$ according to a distribution π. In particular, Markov chain Monte Carlo methods are sampling methods based on Markov chains which have for stationary distributions the ones that we want to sample. The idea is to create x_i solely from x_{i-1} (i. e. a memoryless process). The principle relies on creating the transition matrix P of a time-homogeneous Markov chain such that for any distribution μ,

$$P^k \mu \underset{k\to\infty}{\longrightarrow} \pi \tag{15}$$

so that the distribution converges towards the one that we want to sample. Again, as we just want to give a rough idea, we won't dive into the details of the definition of the convergence or of the criteria ensuring this convergence[6], and we will just be satisfied with knowing that the Metropolis-Hastings algorithm enforces these conditions on the transition matrix of the Markov chain it creates.

3.1.3 Metropolis-Hastings algorithm

For a distribution π known up to a multiplicative constant, the Metropolis-Hastings algorithm creates a Markov chain whose stationary distribution is π. This algorithm is based on the *detailed balance principle*, which is a way to ensure that the generated Markov chain has a unique stationary distribution and converges[7]. The detailed balance equation can be written as:

$$\pi(x)P(x'|x) = \pi(x')P(x|x') \tag{16}$$

where $\pi(x)P(x'|x)$ stands for the probability of the chain being in state x and going to state x' in one step. The key is to split up the transition matrix into two factors corresponding to two sub-steps: when moving from one state x to the other, we first select a new candidate x' with probability $g(x'|x)$[8] and then we accept or reject this candidate with acceptance probability $A(x'|x)$:

$$P(x'|x) = \begin{cases} g(x'|x)A(x'|x) & \text{if } x' \neq x \\ 1 - \sum_{y\neq x} P(y|x) & \text{if } x' = x \end{cases} \tag{17}$$

[6] If the transition matrix P is irreducible aperiodic and positive recurrent, we have the convergence. This is the ergodic theorem. The irreducibility guarantees the existence and uniqueness of the stationary distribution.

[7] To be precise, the transition matrix P as defined below is reversible for π, is irreducible if the selection matrix Q is irreducible, and if the acceptance probability $A < 1$, it P is aperiodic.

[8] corresponding to the selection matrix Q mentioned in the previous footnote.

We impose that $g(x|y) > 0 \Rightarrow g(y|x) > 0$. Thus, the detailed balance condition becomes:

$$\frac{A(x'|x)}{A(x|x')} = \frac{\pi(x')}{\pi(x)} \frac{g(x|x')}{g(x'|x)} \tag{18}$$

So, A has to be chosen as a function of π and g in such a way that the detailed balance is satisfied. The general way to do this is to choose a function $F :]0, \infty[\to]0, 1]$ satisfying

$$F(u) = uF\left(\frac{1}{u}\right) \tag{19}$$

For instance: $F(u) = \min(1, u)$. Then, if

$$A(x'|x) = \begin{cases} F\left(\frac{\pi(x')}{\pi(x)} \frac{g(x|x')}{g(x'|x)}\right) & \text{if } g(x'|x) \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

detailed balance is automatically satisfied. Once one has an acceptance probability A and a selection probability g, the algorithm goes as follows:

1. If $n = 0$: Initialize the state x_0
2. At step n : Generate y_{n+1} with law $g(y_{n+1}|x_n)$
3. Select a number a in $[0, 1]$ randomly with uniform probability
4. If $a < A(y_{n+1}|x_n)$: accept the selected value and set $x_{n+1} = y_{n+1}$. Else, reject the value : $x_{n+1} = x_n$

3.1.4 Relevant examples of the Metropolis-Hastings algorithm

To illustrate the idea of the Metropolis-Hastings algorithm, we will give a few examples of algorithms, based on this principle, which are relevant for spin systems on lattices.

The single-spin-flip algorithm This algorithm allows to study the at-equilibrium thermodynamics of spin systems, i.e. to sample the states according to the Boltzmann weight[9] $\pi(x) = \frac{1}{Z} e^{-\beta E(x)}$ where Z is the partition function. We consider a system of Ising spins on a lattice of N sites. In this algorithm, the idea is to define the selection distribution as follows:

$$g(x'|x) = \begin{cases} 1/N & \text{if } x \text{ and } x' \text{ differ by a single spin flip} \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

Notice that $g(x'|x) = g(x|x')$ in this case. The acceptance probability is then chosen as

$$A(x'|x) = \min\left(1, \frac{\pi(x')}{\pi(x)}\right) = \min\left(1, e^{-\beta(E(x') - E(x))}\right) \tag{22}$$

This algorithm is quite powerful. This is what was used at first to study the dipolar model in [14].

[9] In all of this report we work with $k_B = 1$ and thus $\beta = 1/T$

Parallel tempering When using the single-spin-flip algorithm, there can easily be issues with the thermalisation at low temperatures. An easy way to help the system at low temperature is to simulate various temperatures in parallel and regularly propose another kind of update. The idea is that two states at two neighbouring temperatures get the chance to be flipped. At a given step:

$$g(x'|x) = \begin{cases} 1 & \text{if } x \text{ and } x' \text{ are states in neighbouring temperature threads} \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

In this case, considering the joint distributions of the states in the two different temperature threads β_a, β_b, the acceptance probability is given by:

$$A(x' = (x_2, x_1)|x = (x_1, x_2)) = \min\left\{1, e^{(\beta_a - \beta_b)(E(x_1) - E(x_2))}\right\} \tag{24}$$

Cluster algorithms and dual worm algorithms We won't go into the details of such algorithms[10], since one example of dual worm update will be seen in the following. But let us mention that these algorithms create non-local updates which are needed to fight against the critical slowing down (the drastic increase in the relaxation time of the Metropolis dynamics) near phase transitions. This slowing down is an issue as it makes it almost impossible to make independent measurements in a reasonable amount of time. Proposing cluster updates, i.e. flipping clusters of spins at once, corresponds to changing the transition matrix of the Markov chain and reduces drastically the number of steps needed to get two independent measurements. As we will see in the next sections, dual worm algorithms are a way of creating such clusters, based on a mapping of the spin model onto a dimer model on the dual lattice.

3.2 The myopic algorithm

In their article, G. Rakala and K. Damle present two rejection-free[11] dual worm algorithms for the simulation of frustrated two dimensional Ising antiferromagnets. Although they focus mainly on the triangular lattice with spin-spin interactions extending up to 3$^{\text{rd}}$ neighbours, they mention that one of these algorithms can be directly generalized to the case of the Ising antiferromagnet on the kagome lattice with up to 3$^{\text{rd}}$ neighbour spin-spin interactions. *A priori*, as will be shown later on, we think that their algorithm can even be readily generalized to even longer-range couplings on both lattices.

3.2.1 General idea

As we introduced in section 3.1.4, the idea of a cluster algorithm is to build global updates of the spin systems that can be accepted with a relatively high probability under the detailed balance condition. The dual worm approach is a way to build such clusters by working on the dual of the lattice. In order to try and improve the efficiency of the algorithm, instead of building the worm update first and then accepting or rejecting it, let us see a way to build a worm update which automatically satisfies the detailed balance condition [15].

[10] And we won't present the Wolff algorithm nor the Swendsen-Wang algorithm.
[11] i.e. where detailed balance is enforced without using the acceptance probability.

References

[1] I. Syôzi, Prog. Theor. Phys. **6**, 306 (1951)

[2] K. Kanô and S. Naya, Prog. Theor. Phys. **10**, 158 (1953)

[3] A. Sütö, Z. Phys. B **44**, 121 (1981)

[4] A. Smerald, F. Mila, Private Communication

[5] C. Nisoli, R. Moessner and P. Schiffer, Rev. Mod. Phys. **85**, 1473 (2013)

[6] R. F. Wang *et al.*, Nature (London) **439**, 303 (2006)

[7] G. Möller and R. Moessner, Phys. Rev. Lett. **96**, 237202 (2006)

[8] A. Farhan *et al.*, Nat. Phys. **9**, 375 (2013)

[9] A. Farhan *et al.*, Phys. rev. Lett. **111**, 057204 (2013)

[10] L. Anghinolfi *et al.*, Nat. Commun. **6**, 8278 (2015)

[11] G.-W. Chern, P. Mellado and O. Tchernyshyov, Phys. Rev. Lett. **106**, 207202 (2011)

[12] S. Zhang *et al.*, Phys. Rev. Lett. **109**, 087201 (2012)

[13] I. A. Chioar *et al.*, Phys. Rev. B **90**, 064411 (2014)

[14] I. A. Chioar, N. Rougemaille, B. Canals, Phys. Rev. B **93**, 214410 (2016)

[15] G. Rakala and K. Damle, arXiv:1612.00851v1 (2016)

[16] G. H. Wannier, Phys. Rev. **79**, 356 (1950)

[17] J. Stephenson, J. Math. Phys. **5**, 1009 (1964)

[18] T. Takagi and M. Mekata, J. Phys. Soc. Jpn. **62**, 11 3943 (1993)

[19] R. G. Melko, B. C. den Hertog and M. J. P. Gingras, Phys. Rev. Lett. **87**, 067203 (2001)

[20] A. W. Sandvik, arXiv:1101.3281 (2011)

[21] M. Benaïm and N. El Karoui, Promenade aléatoire: chaînes de Markov et simulations; martingales et stratégies, (Les Editions de l'Ecole Polytechnique, Collection: Mathématiques Appliquées, Palaiseau, 2006)

[22] O. F. Syljuåsen and A. W. Sandvik, Phys. Rev. E **66**, 046701 (2002)

[23] O. F. Syljuåsen, Phys. Rev. E **67**, 046701 (2003)