Algorithms and computations in physics (Oxford Lectures 2025)

Werner Krauth*
Laboratoire de Physique, Ecole normale supérieure, Paris (France)
Rudolf Peierls Centre for Theoretical Physics
University of Oxford (UK)

Eighth lecture: 11 March 2025
(version: 03/11/2025)

The final lecture in the set is on the classical Ising model. We go from basic (and not so basic) enumerations to the high-temperature expansion of the Ising model, touch on its exact solution by Kac and Ward (1952) and then treat sampling methods: Metropolis, heatbath, and perfect sampling, to finish with the Wolff (1989) cluster algorithm.

Contents

8 The Ising model: From the Gray code to cluster algorithms

The Ising model describes spins $\sigma_k \pm 1$, (or $+/-$) for $k = 1, \ldots, N$, on a lattice, for example a two-dimensional square lattice. In the simplest case, the ferromagnetic Ising model, neighboring spins prefer to align. This means that pairs $\{+,+\}$ and $\{-,-\}$ of neighboring spins direction have a lower energy than antiparallel spins (pairs $\{+,-\}$ and $\{-,+\}$), as expressed by the energy

$$E = -J \sum_{\langle k,l \rangle} \sigma_k \sigma_l. \tag{8.1}$$

*werner.krauth@ens.fr, werner.krauth@physics.ox.ac.uk

The sum is over all pairs of neighbors and, by convention, each edge (pair of neighbors) $\langle k, l\rangle = \langle l, k\rangle$ is counted only once. The parameter J is positive (and identical for all pairs k, l), and we may take it equal to one. We consider the partition function

$$Z = \sum_{\{\sigma_1,\ldots,\sigma_N\}=\{\pm1,\ldots,\pm1\}} \exp\left(-\beta E\right) \tag{8.2}$$

that we will, on the one hand, evaluate and, on the other hand, sample.

procedure `energy-ising`
input $\{\sigma_1, \ldots, \sigma_N\}$
$E \leftarrow 0$
for $k = 1, \ldots, N$**:**
  for $n = 1, \ldots, d$**:** (d: space dimension, sum over half of nbrs)
    $j \leftarrow \mathrm{Nbr}(n, k)$
    if $(j \neq 0)$ **then**
      $E \leftarrow E - \sigma_k \sigma_j$
output E

Algorithm 8.1: `energy-ising`. Computing the energy of an Ising-model configuration. In order to avoid overcounting, $\mathrm{Nbr}(., k)$ goes only over half the neighbors of k.

8.1 Enumeration: listing and counting

Listing and counting are two aspects of the word *enumeration*. In this section, we illustrate both aspects.

8.1.1 Spin enumerations: Naive and minimal-change order (Gray code)

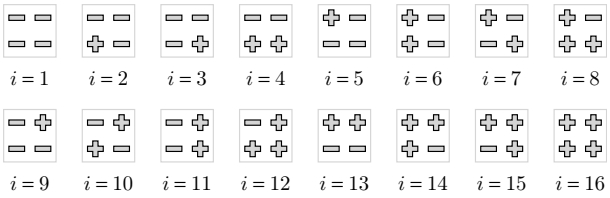

Figure 8.1: List of configurations of the Ising model on a 2×2 square lattice.

In Fig. 8.1, we list all configurations for a (magnetic) Ising model on a 2×2 lattice. Without periodic boundary conditions, configurations $i = 1$ and $i = 16$ have an energy $E = -4$, and configurations $i = 7$ and $i = 10$ have an energy $E = +4$. All others are zero-energy configurations.

In the present subsection, we enumerate all the spin configurations of the Ising model; in fact, we list them one after another. Most simply, each configuration $i = 1, \ldots, 2^N$ of N Ising spins is related to the binary representation of the number $i - 1$: in Fig. 8.1, zeros in the binary representation of $i-1$ correspond to down spins, and ones to up spins. As an example, the binary representation of the decimal number 10 (configuration $i = 11$ in Fig. 8.1) is 1010, which yields a spin configuration $\{+, -, +, -\}$ to be translated to the lattice with our standard numbering scheme. In Python, one uses the function `bin` or one of its variants.

It is faster to compute the change of energy resulting from a spin-flip rather than the energy itself. In Fig. 8.2, for example, we can find out that $E_b = E_a - 4$, simply because the "molecular

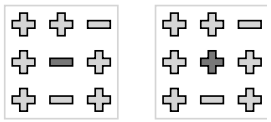

Figure 8.2: Two configurations of the Ising model connected by the flip of a single spin.

field" acting on the central site equals 2 (it is generated by three up spins and one down spin). The change in energy is equal to twice the value of the spin at the site times the molecular field.

procedure `gray-flip`
input $\{\tau_0, \ldots, \tau_N\}$
$k \leftarrow \tau_0$
if $k > N$**:** **exit**
$\tau_{k-1} \leftarrow \tau_k$
$\tau_k \leftarrow k + 1$
if $k \neq 1$**:** $\tau_0 \leftarrow 1$
output $k, \{\tau_0, \ldots, \tau_N\}$

Algorithm 8.2: `gray-flip`. Gray code for spins $\{1, \ldots, N\}$. k is the next spin to flip. Initially, $\{\tau_0, \ldots, \tau_N\} = \{1, \ldots, N+1\}$.

On lattices of any size, the change in energy can be computed in a constant number of operations, whereas the effort for calculating the energy grows with the number of edges. Therefore it is interesting that all 2^N spin configurations can be enumerated through a sequence of 2^N spin-flips, one at a time. (Equivalently, one may enumerate all numbers $\{0, \ldots, 2^N - 1\}$ by changing a single digit at a time during the enumeration.) Algorithms that perform such enumerations are called Gray codes, and an application of a Gray code for four spins is shown in Table 8.1.1. How it works can be understood by (mentally) folding Table 8.1.1 along the horizontal line between configurations $i = 8$ and $i = 9$: the configurations of the first three spins $\{\sigma_1, \sigma_2, \sigma_3\}$ are folded onto each other (the first three spins are the same for $i = 8$ and $i = 9$, and also for $i = 7$ and $i = 10$, etc.). The spins $\{\sigma_1, \sigma_2, \sigma_3\}$ remain unchanged between $i = 8$ and $i = 9$, and this is the only moment at which σ_4 flips, namely from $-$ to $+$. To write down the Gray code for $N = 5$, we would fold Table 8.1.1 along the line following configuration $i = 16$, and insert $\{\sigma_5(i = 1), \ldots, \sigma_5(i = 16)\} = \{-, \ldots, -\}$, and $\{\sigma_5(i = 17), \ldots, \sigma_5(i = 32)\} = \{+, \ldots, +\}$. Algorithm 8.2 (`gray-flip`) provides a practical implementation. We may couple the Gray code enumeration to an update of the energy (see Alg. 8.3 (`enumerate-ising`)). Of course, the Gray code still has exponential running time, but the enumeration as in Fig. 8.3 gains a factor $\propto N$ with respect to naive binary enumeration.

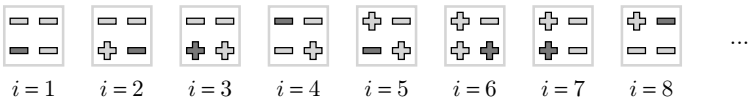

Figure 8.3: List of Ising-model configurations on a 2×2 square lattice, generated by the Gray code (only the dark spins flip, see Table **??**).

Algorithm 8.3 (`enumerate-ising`) does not directly compute the partition function at inverse temperature β, but rather the number of configurations with energy E, in other words, the density of states $\mathcal{N}(E)$ (see Table 8.1.1). We must take care in implementing this program

i	$\{\sigma_1, \ldots, \sigma_4\}$			
1	−	−	−	−
2	+	−	−	−
3	+	+	−	−
4	−	+	−	−
5	−	+	+	−
6	+	+	+	−
7	+	−	+	−
8	−	−	+	−
9	−	−	+	+
10	+	−	+	+
11	+	+	+	+
12	−	+	+	+
13	−	+	−	+
14	+	+	−	+
15	+	−	−	+
16	−	−	−	+

Table 8.1: Gray-code enumeration of spins $\{\sigma_1, \ldots, \sigma_4\}$. Each configuration differs from its predecessor by one spin only.

because $\mathcal{N}(E)$ can easily exceed 2^{31}, the largest integer that fits into a standard four-byte computer word. We note, in our case, that it suffices to generate only half of the configurations, because $E(\sigma_1, \ldots, \sigma_N) = E(-\sigma_1, \ldots, -\sigma_N)$.

8.1.2 Loop-listings in the Ising model

As mentioned, the word "enumeration" may refer to listing items (configurations), but also to simply counting them. In the list generated by Alg. 8.3 (`enumerate-ising`), we were able to pick out any information we wanted, for example the number of configurations of energy E and magnetization M, that is, the density of states $\mathcal{N}(E, M)$. In this subsection we discuss an alternative enumeration for the Ising model. It enumerates loop configurations which appear in the high-temperature expansion of the Ising model. This program will then turn, in Sec. 8.1.3, into an enumeration of the second kind (Kac and Ward, 1954), which sums up all loops and obtains the exact $Z(\beta)$ for a two-dimensional Ising system of any size (Kaufman, 1949), and even for the infinite system (Onsager, 1944). However, it then counts without listing. For example, it finds the number $\mathcal{N}(E)$ of configurations with energy E but does not tell us how many of them have a magnetization M.

The story starts with Van der Waerden who, in 1941, noticed that the Ising-model partition function,

$$Z = \sum_{\sigma} \exp\Big(J\beta \sum_{\langle k,l \rangle} \sigma_k \sigma_l\Big) = \sum_{\sigma} \prod_{\langle k,l \rangle} \mathrm{e}^{J\beta\sigma_k\sigma_l}, \tag{8.3}$$

allows each term $\mathrm{e}^{J\beta\sigma_k\sigma_l}$ to be expanded and rearranged into just two terms, one independent

procedure `enumerate-ising`
$\{\mathcal{N}(-2N), \ldots, \mathcal{N}(2N)\} \leftarrow \{0, \ldots, 0\}$
$\{\sigma_1, \ldots, \sigma_N\} \leftarrow \{-1, \ldots, -1\}$
$\{\tau_0, \ldots, \tau_N\} \leftarrow \{1, \ldots, N+1\}$
$E \leftarrow -2N$
$\mathcal{N}(E) \leftarrow 2$
for $i = 1, \ldots, 2^{N-1} - 1$**:**
$\left\{ \begin{array}{l} k \leftarrow \texttt{gray-flip}\,(\{\tau_0, \ldots, \tau_N\}) \\ h \leftarrow \sum_{l=1}^{2d} \sigma_{\mathrm{Nbr}(l,k)} \text{ (field on site } k) \\ E \leftarrow E + 2 \cdot \sigma_k h \\ \mathcal{N}(E) \leftarrow \mathcal{N}(E) + 2 \\ \sigma_k \leftarrow -\sigma_k \end{array} \right.$
output $\{\mathcal{N}(E) > 0\}$

Algorithm 8.3: `enumerate-ising`. Single spin-flip (Gray code) enumeration for the Ising model, using Alg. 8.2 (`gray-flip`).

of the spins and the other proportional to $\sigma_k\sigma_l$:

$$\mathrm{e}^{\beta\sigma_k\sigma_l} = 1 + \beta\sigma_k\sigma_l + \frac{\beta^2}{2!}\underbrace{(\sigma_k\sigma_l)^2}_{=1} + \frac{\beta^3}{3!}\underbrace{(\sigma_k\sigma_l)^3}_{=\sigma_k\sigma_l} + \cdots + \cdots$$
$$= \underbrace{\left(1 + \frac{\beta^2}{2!} + \frac{\beta^4}{4!} + \cdots\right)}_{\cosh\beta} + \sigma_k\sigma_l \underbrace{\left(\beta + \frac{\beta^3}{3!} + \frac{\beta^5}{5!} + \cdots\right)}_{\sinh\beta}$$
$$= (\cosh\ \beta)\,(1 + \sigma_k\sigma_l \tanh\ \beta)\,.$$

Inserted into eq. (8.3), with $J = +1$, this yields

$$Z(\beta) = \sum_{\boldsymbol{\sigma}} \prod_{\langle k,l\rangle} ((\cosh\ \beta)\,(1 + \sigma_k\sigma_l \tanh\ \beta))\,. \tag{8.4}$$

For concreteness, we continue with a 4×4 square lattice without periodic boundary conditions. This lattice has 24 edges and 16 sites, so that, by virtue of eq. (8.4), its partition function $Z_{4\times4}(\beta)$ is the product of 24 parentheses, one for each edge:

$$Z_{4\times4}(\beta) = \sum_{\{\sigma_1,\ldots,\sigma_{16}\}} \cosh^{24}\beta \overbrace{(1 + \sigma_1\sigma_2 \tanh\ \beta)}^{\text{edge 1}} \overbrace{(1 + \sigma_1\sigma_5 \tanh\ \beta)}^{\text{edge 2}}$$
$$\times \ldots (1 + \sigma_{14}\sigma_{15} \tanh\ \beta) \underbrace{(1 + \sigma_{15}\sigma_{16} \tanh\ \beta)}_{\text{edge 24}}. \tag{8.5}$$

We multiply out this product: for each edge (parenthesis) k, we have a choice between a "one" and a "tanh" term. This is much like the option of a spin-up or a spin-down in the original Ising-model enumeration, and can likewise be expressed through a binary variable n_k:

$$n_k = \begin{cases} 0 & (\equiv \text{ edge } k \text{ in eq. (8.5) contributes 1}) \\ 1 & (\equiv \text{ edge } k \text{ contributes } (\sigma_{s_k}\sigma_{s'_k} \tanh\beta)) \end{cases},$$

where s_k and s'_k indicate the sites at the two ends of edge k. Edge $k = 1$ has $\{s_1, s'_1\} = \{1, 2\}$,

	$\mathcal{N}(E) = \mathcal{N}(-E)$		
E	2×2	4×4	6×6
0	12	20 524	13 172 279 424
4	0	13 568	11 674 988 208
8	2	6 688	8 196 905 106
12	.	1 728	4 616 013 408
16	.	424	2 122 173 684
20	.	64	808 871 328
24	.	32	260 434 986
28	.	0	71 789 328
32	.	2	17 569 080
36	.	.	3 846 576
40	.	.	804 078
44	.	.	159 840
48	.	.	35 148
52	.	.	6 048
56	.	.	1 620
60	.	.	144
64	.	.	72
68	.	.	0
72	.	.	2

Table 8.2: Density of states $\mathcal{N}(E)$ for small square lattices with periodic boundary conditions (from Alg. 8.3 (`enumerate-ising`))

and edge $k = 24$ has, from eq. (8.5), $\{s_{24}, s'_{24}\} = \{15, 16\}$. Each factored term can be identified by variables

$$\{n_1, \ldots, n_{24}\} = \{\{0, 1\}, \ldots, \{0, 1\}\}.$$

For $\{n_1, \ldots, n_{24}\} = \{0, \ldots, 0\}$, each parenthesis picks a "one". Summed over all spin configurations, this gives 2^{16}. Most choices of $\{n_1, \ldots, n_{24}\}$ average to zero when summed over spin configurations because the same term is generated with $\sigma_k = +1$ and $\sigma_k = -1$. Only choices leading to spin products $\sigma_s^0, \sigma_s^2, \sigma_s^4$ at each lattice site s remain finite after summing over all spin configurations. The edges of these terms form loop configurations, such as those shown for the 4×4 lattice in Fig. 8.4. The list of all loop configurations may be generated by Alg. 8.4 (`edge-ising`), a recycled version of the Gray code for 24 digits, coupled to an incremental calculation of the number of spins on each site. The $\{o_1, \ldots, o_{16}\}$ count the number of times the sites $\{1, \ldots, 16\}$ are present. The numbers in this vector must all be even for a loop configuration, and for a nonzero contribution to the sum in eq. (8.5).

For the thermodynamics of the 4×4 Ising model, we only need to keep track of the number of edges in each configuration, not the configurations themselves. Table 8.1.2, which shows the number of loop configurations for any given number of edges, thus yields the exact partition function for the 4×4 lattice without periodic boundary conditions:

$$Z_{4\times4}(\beta) = \left(2^{16} \cosh^{24} \beta\right) \left(1 + 9 \tanh^4 \beta + 12 \tanh^6 \beta + \cdots + 4 \tanh^{18} \beta + 1 \tanh^{20} \beta\right). \quad (8.6)$$

Partition functions obtained from this expression are easily checked against the Gray-code enumeration (if we remember to use the same boundary conditions). Up to a certain power of β, this approach corresponds to a high-temperature expansion, rather than an enumeration.

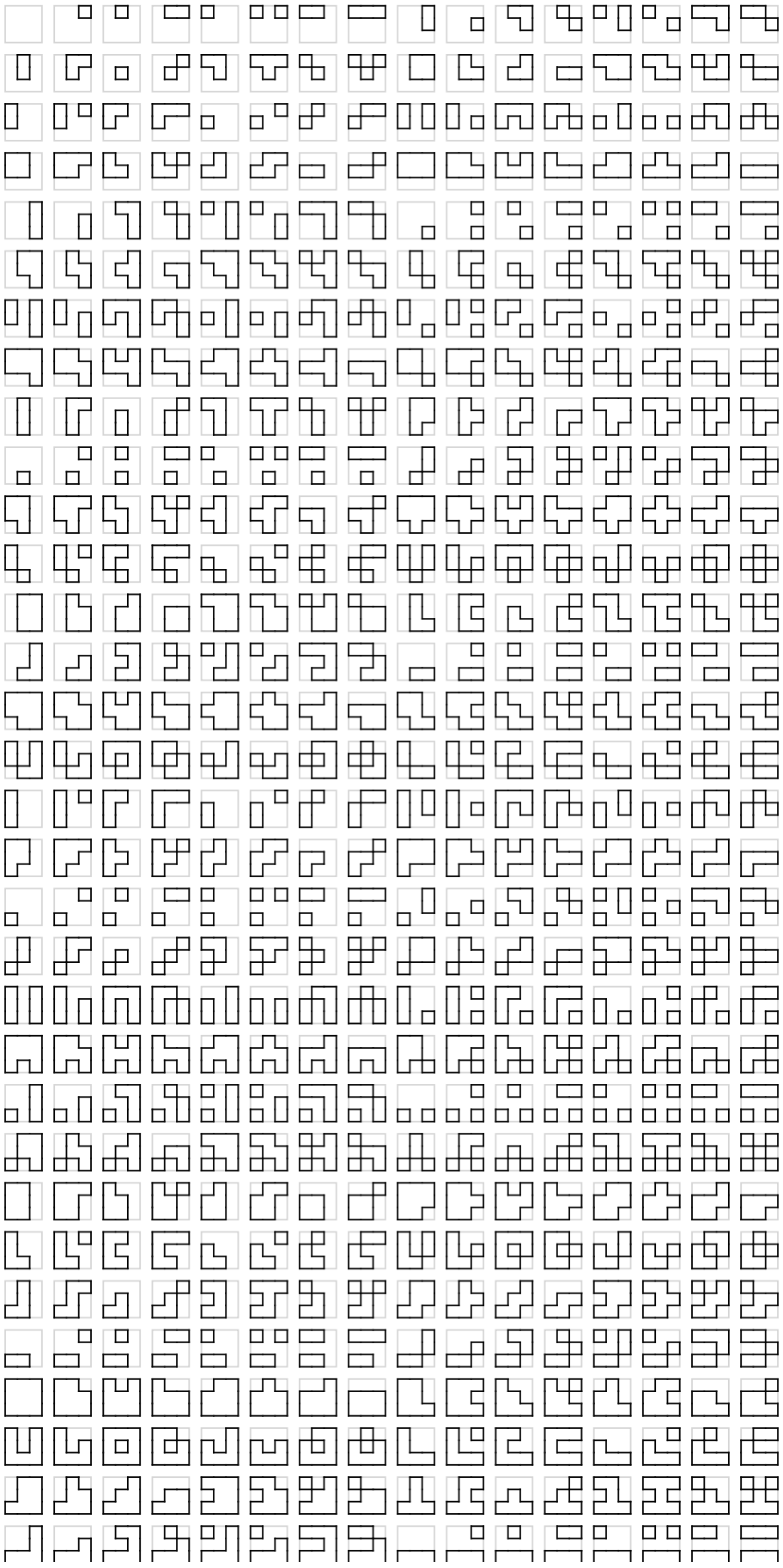

Figure 8.4: The list of all 512 loop configurations for the 4×4 Ising model without periodic boundary conditions (from Alg. 8.4 (`edge-ising`)).

# Edges	# Configs
0	1
4	9
6	12
8	50
10	92
12	158
14	116
16	69
18	4
20	1

Table 8.3: Numbers of loop configurations in Fig. 8.4 with given numbers of edges (the figure contains one configuration with 0 edges, 9 with 4 edges, etc). (From Alg. 8.4 (`edge-ising`)).

procedure `edge-ising`
input $\{(s_1, s'_1), \ldots, (s_{24}, s'_{24})\}$
$\{n_1, \ldots, n_{24}\} \leftarrow \{0, \ldots, 0\}$
$\{\tau_0, \ldots, \tau_{24}\} \leftarrow \{1, \ldots, 25\}$
$\{o_1, \ldots, o_{16}\} \leftarrow \{0, \ldots, 0\}$
output $\{n_1, \ldots, n_{24}\}$
for $i = 1, 2^{24} - 1$**:**
$k \leftarrow$ `gray-flip` $(\{\tau_0, \ldots, \tau_{24}\})$
$n_k \leftarrow$ `mod` $(n_k + 1, 2)$
$o_{s_k} \leftarrow o_{s_k} + 2 \cdot n_k - 1$
$o_{s'_k} \leftarrow o_{s'_k} + 2 \cdot n_k - 1$
if ($\{o_1, \ldots, o_{16}\}$ all even) **then**
{ **output** $\{n_1, \ldots, n_{24}\}$

Algorithm 8.4: `edge-ising`. Gray-code enumeration of the loop configurations in Fig. 8.4. The edge k connects neighboring sites σ_k and σ'_k.

8.1.3 Counting beyond listing

We do not have time here to elaborate fully (see Ref. [1] for more details), but the high-temperature sum over loops is closely related to the exact solution of the two-dimensional Ising model by Onsager (1947), in the formulation of Kac and Ward (1952). That solution is based on the representation of a determinant of a matrix in terms of the permutation cycles that we discussed in Lecture 7.

This is possible because the determinant of a matrix $U = (u_{kl})$ is defined by a sum of permutations P (with signs and weights). As we discussed in Lecture 7, each permutation may be written as a collection of cycles, a "cycle configuration". Our task will consist in choosing the elements u_{kl} of the matrix U in such a way that the signs and weights of each cycle configurations correspond to the loop configurations in the two-dimensional Ising model. This makes us arrive at a computer program which implements the correspondence, and effectively solves the enumeration problem for large two-dimensional lattices. To give an idea of how this is done, we restrict ourselves to square lattices without periodic boundary conditions, and consider the definition of the determinant of a matrix U,

$$\det U = \sum_{\text{permutations}} (\text{sign}\, P) u_{1P_1} u_{2P_2} \ldots u_{NP_N}.$$

We now represent P in terms of cycles, just as we did in Lecture 7. The sign of a permutation P of N elements with n cycles is $\operatorname{sign} P = (-1)^{N+n}$ (see Ref. [1] for a discussion). For matrices with even N, for which $\operatorname{sign} P = (-1)^{\#\text{ of cycles}}$. The determinant for even N is thus

$$\det U = \sum_{\substack{\text{cycle}\\ \text{configs}}} (-1)^{\#\text{ of cycles}} \underbrace{u_{P_1P_2}u_{P_2P_3}\dots u_{P_MP_1}}_{\text{weight of first cycle}} \underbrace{u_{P'_1P'_2}\dots}_{\text{other cycles}}$$

$$= \sum_{\substack{\text{cycle}\\ \text{configs}}} \left(\left\{\begin{matrix}(-1)\cdot \text{ weight of}\\ \text{first cycle}\end{matrix}\right\}\right) \times \dots \times \left(\left\{\begin{matrix}(-1)\cdot \text{ weight of}\\ \text{last cycle}\end{matrix}\right\}\right).$$

This representation of a determinant in terms of cycle configurations suggests that we should choose the matrix elements u_{kl} such that each cycle corresponding to a loop on the lattice (for example $(P_1, \dots, P_M)$) gets a negative sign (this means that the sign of $u_{P_1P_2}u_{P_2P_3}\dots u_{P_MP_1}$ should be negative). All cycles not corresponding to loops should get zero weight. That this was at all possible was the good fortune of Kac and Ward. Their work is also discussed in the statistical mechanics book by Feynman [2]. (The final lecture notes will contain an implementation of a $4N \times 4N$ matrix where the square root of the determinant gives the partition function.)

8.2 Single-spin Monte Carlo updates in the Ising model

Gray-code enumerations, as discussed in Sec. 8.1, succeed only for relatively small systems, and high-temperature expansions, the subject of Sec. 8.1.3, must usually be stopped after a limited number of terms, before they get too complicated. Only under exceptional circumstances, as in the two-dimensional Ising model, can these methods be pushed much further. Often, Monte Carlo methods alone are able to obtain exact results for large sizes. The price to be paid is that configurations are sampled rather than enumerated, so that statistical errors are inevitable. The Ising model has been a major test bed for Monte Carlo simulations and algorithms of all kinds, and its theoretical analysis has been pushed to a very high level.

8.2.1 Metropolis algorithm

A basic task in statistical physics is to write a local Metropolis algorithm for the Ising model. This program is even simpler than the Markov-chain algorithm for hard disks that we studied in Lecture 4. In addition, irreducibility and aperiodicity are easy to establish. The Ising model has no immediate connection with classical mechanics (there is no molecular dynamics algorithm for the model). Analogously to Alg. `(markov-disks)`, we randomly pick a site and attempt to flip the spin at that site (see Fig. 8.5).

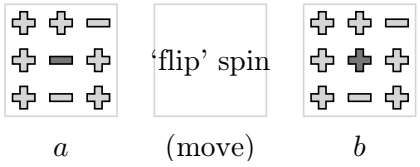

Figure 8.5: Local Monte Carlo move $a \to b$ in the Ising model, to be accepted with probability $\min\left[1, \mathrm{e}^{-\beta(E_b-E_a)}\right]$.

The proposed move between configuration a, with energy E_a, and configuration b, with energy E_b, must be accepted with probability $\min\left\{1, \mathrm{e}^{-\beta(E_b-E_a)}\right\}$, as straightforwardly implemented in Alg. 8.5 (`markov-ising`). We can check the results on small lattices against exact enumeration, before moving on to larger ones, but will find that the algorithm is very slow.

procedure `markov-ising`
input $\{\sigma_1, \ldots, \sigma_N\}, E$
$k \leftarrow \texttt{nran}\,(1, N)$
$h \leftarrow \sum_{l=1}^{2d} \sigma_{\mathrm{Nbr}(l,k)}$
$\Delta_E \leftarrow 2h\sigma_k$
$\Upsilon \leftarrow \texttt{ran}(0, 1)$
if $(\Upsilon < \exp\,(-\beta\Delta_E)\,)$ **then**
$\begin{cases} \sigma_k \leftarrow -\sigma_k \\ E \leftarrow E + \Delta_E \end{cases}$
output $\{\sigma_1, \ldots, \sigma_N\}, E$

Algorithm 8.5: `markov-ising`. Local Metropolis algorithm for the Ising model in d dimensions.

In Alg. 8.5 (`markov-ising`), the spin to be updated was chosen randomly. This defines a reversible algorithm, for which it is easy to establish the detailed-balance condition. Sequential updates will make the algorithm non-reversible, but we can prove that the global-balance condition is still satisfied. It appears that any reversible algorithm can be sequentialized in this way, but this is already the point where we transition from an area where everything is understood into one where little is known.

8.2.2 Heat-bath algorithm

We discuss an alternative to the Metropolis Monte Carlo method, the heat-bath algorithm. Rather than flipping a spin at a random site, we now thermalize this spin with its local environment (see Fig. 8.6). In the presence of a molecular field h at site k, the spin points up and down with probabilities π_h^+ and π_h^-, respectively, where

$$\begin{aligned} \pi_h^+ &= \frac{\mathrm{e}^{-\beta E^+}}{\mathrm{e}^{-\beta E^+} + \mathrm{e}^{-\beta E^-}} = \frac{1}{1 + \mathrm{e}^{-2\beta h}}, \\ \pi_h^- &= \frac{\mathrm{e}^{-\beta E^-}}{\mathrm{e}^{-\beta E^+} + \mathrm{e}^{-\beta E^-}} = \frac{1}{1 + \mathrm{e}^{+2\beta h}}. \end{aligned} \tag{8.7}$$

These probabilities are normalized ($\pi_h^+ + \pi_h^- = 1$), and π_h^+ is an increasing function of h. To

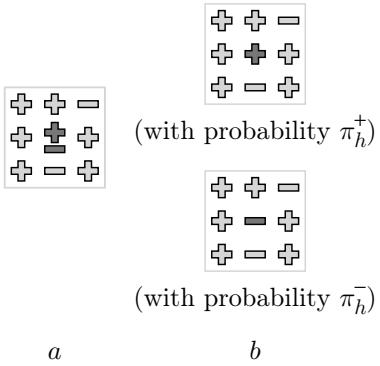

Figure 8.6: Heat bath algorithm for the Ising model. The spin on the central site has a molecular field $h = 2$ (see Alg. 8.6 (`heatbath-ising`)).

sample the Bernoulli distribution defined in eq. (8.7), we pick a random number $\Upsilon = \texttt{ran}(0, 1)$ and let the spin point up if $\Upsilon < \pi_h^+$ and down otherwise. The action taken is independent of the spin's orientation before the move (see Alg. 8.6 (`heatbath-ising`)). The heat bath algorithm

is local, just as the Metropolis algorithm, and its performance is essentially the same. The algorithm is conveniently represented in a diagram of the molecular field h against the random number Υ (see Fig. 8.7).

procedure `heatbath-ising`
input $\{\sigma_1, \ldots, \sigma_N\}, E$
$k \leftarrow$ `nran` $(1, N)$
$h \leftarrow \sum_n \sigma_{\mathrm{Nbr}(n,k)}$
$\sigma' \leftarrow \sigma_k$
$\Upsilon \leftarrow$ `ran`$(0, 1)$
if $(\Upsilon < \pi_h^+)$ **then** (see eq. (8.7))
  $\{\ \sigma_k \leftarrow 1$
else:
  $\{\ \sigma_k \leftarrow -1$
if $\sigma' \neq \sigma_k$**:** $E \leftarrow E - 2h\sigma_k$
output $\{\sigma_1, \ldots, \sigma_N\}$

Algorithm 8.6: `heatbath-ising`. Heat bath algorithm for the Ising model.

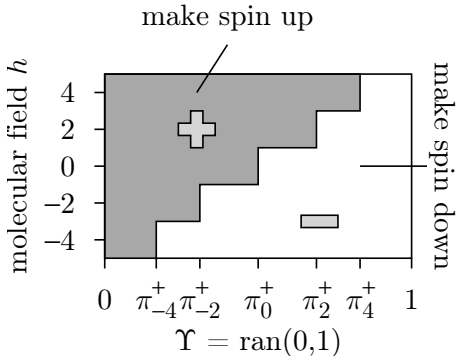

Figure 8.7: Action to be taken in the heat bath algorithm as a function of the molecular field h and the random number Υ.

A Python implementation of this fundamental algorithm can be found here.

8.2.3 Perfect sampling

We remember from earlier lectures that Markov-chain algorithms, although the distribution they sample is equal to the stationary distribution π (the Boltzmann distribution of eq. (8.2), in our context of the Ising model), show the phenomenon of coupling, that can (sometimes) be crafted into perfect-sampling algorithms. This is what we want to do now.

Let us first consider the Alg. 8.6 (`heatbath-ising`), but not for one configuration, but for two configurations $\{\sigma_1, \ldots, \sigma_N\}$ and $\{\tau_1, \ldots, \tau_N\}$ with $s_l \leq \tau_l \; \forall l$ (see Alg. 8.7 (`heatbath-two-ising`)). Using the same random numbers to generate the spin k to flip, and the random number Υ, we notice that, after one step, the ordering between the two configurations is preserved.

We could now go on and write a last program Alg. (`heatbath-three-ising`), with *three* configurations, say, $\{\rho_1, \ldots, \rho_N\}$, $\{\sigma_1, \ldots, \sigma_N\}$, and $\{\tau_1, \ldots, \tau_N\}$ with, initially, $\rho_k \leq \sigma_k \leq \tau_k$. It suffices to start off this program with $\rho_k = -1$, and $\tau_k = +1$, to see that these extreme configurations (that, luckily, can be run with Alg. 8.7 (`heatbath-two-ising`)) "herd in" all other configurations, and that once σ and τ have coupled, all 2^N initial configurations have

procedure `heatbath-two-ising`
input $\{\sigma_1, \ldots, \sigma_N\}$, $\{\tau_1, \ldots, \tau_N\}$ (we suppose $\sigma_l \leq \tau_l \ \forall l$)
$k \leftarrow \texttt{nran}\,(1, N)$
$h_\sigma \leftarrow \sum_{n=1}^{2d} \sigma_{\mathrm{Nbr}(n,k)}$, $h_\tau \leftarrow \sum_{n=1}^{2d} \tau_{\mathrm{Nbr}(n,k)}$ (NB: $h_\sigma \leq h_\tau$.)
$\Upsilon \leftarrow \texttt{ran}(0, 1)$
if $(\Upsilon < \pi^+_{h_\sigma} \leq \pi^+_{h_\tau})$ **then** (see eq. (8.7))
$\{\ \sigma_k \leftarrow 1,\ \tau_k \leftarrow 1$
else if $\pi^+_{h_\sigma} < \Upsilon < \pi^+_{h_\tau}$**: then**
$\{\ \sigma_k \leftarrow -1,\ \tau_k \leftarrow 1$
else:
$\{\ \sigma_k \leftarrow -1,\ \tau_k \leftarrow -1$
output $\{\sigma_1, \ldots, \sigma_N\}$, $\{\tau_1, \ldots, \tau_N\}$ (NB: $\sigma_l \leq \tau_l \ \forall l$)

Algorithm 8.7: `heatbath-two-ising`. Heat bath algorithm for two configurations in the Ising model. The order between $\{\sigma_1, \ldots, \sigma_N\}$ and $\{\tau_1, \ldots, \tau_N\}$ is preserved.

done also. One easily transforms this algorithm into a coupling-from-the-past algorithm, which samples perfect configurations from the Boltzmann distribution without any approximation [1], for any Ising model with ferromagnetic interactions, at any temperature, and in any dimension. The approach was pioneered by Propp and Wilson [3] in 1996.

8.3 Cluster algorithms

Algorithm 8.5 (`markov-ising`) and its variants, the classic simulation methods for spin models, have gradually given way to cluster algorithms, which converge much faster. These algorithms feature large-scale moves. In the imagery of the heliport game, they propose and accept displacements on the scale of the system, rather than walk about the landing pad in millimeter-size steps. We now discuss cluster methods in a language stressing the practical aspect of a priori probabilities, leading to a pseudocode program and a Python implementation.

We recall that single-spin-flip Monte Carlo algorithms are slow close to T_c, because the histogram of essential values of the magnetization is wide and the step width of the magnetization is small. To sample faster, we must foster moves which change the magnetization by more than ± 2. However, using the single-spin-flip algorithm in parallel, on several sites at a time, only hikes up the rejection rate. Neither can we, so to speak, solidly connect all neighboring spins of the same orientation and flip them all at once. Doing so would quickly lead to a perfectly aligned state, from which there would be no escape.

8.3.1 Wolff cluster algorithm (Ising model)

Let us analyze a more sophisticated rule for flipping spins. We suppose that, starting from a random initial spin, a cluster is constructed by adding, with probability p, neighboring sites with spins of the same orientation. For the moment, this probability is an arbitrary parameter. The above solid connection between neighboring spins corresponds to $p = 1$. During the cluster construction, we keep a list of cluster sites, but also one containing pocket sites, that is, new members of the cluster that can still make the cluster grow. The cluster construction algorithm picks one pocket site and removes it from the pocket. It then checks all of this site's neighbors outside the cluster with spins of like sign and adds these neighbors, with probability p, to the pocket and the cluster (see Fig. 8.8). After completion of the construction of the cluster, when the pocket is empty, all spins in the cluster are flipped. This brings us from the initial configuration

[1] . . . if perfect random numbers are available

a to the final configuration b (see Fig. 8.9). From our experience with a priori probabilities, we know beforehand that a suitable acceptance rule will ensure detailed balance between a and b, for any $0 < p < 1$. In going from a to b, the a priori construction probabilities $\mathcal{A}(a)$ and $\mathcal{A}(b)$,

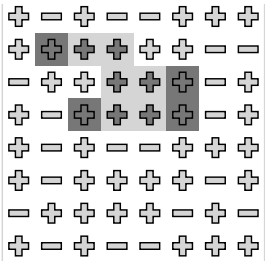

Figure 8.8: Ising configuration with 10 cluster sites (the dark *and* the light gray sites). The dark sites are pocket sites.

the acceptance probabilities $P(a \to b)$ and $P(b \to a)$, and the Boltzmann weights $\pi(a)$ and $\pi(b)$, must respect the detailed-balance condition generalized to the presence of *a priori* probabilities, as discussed in earlier lectures:

$$\pi(a)\mathcal{A}(a \to b)P(a \to b) = \pi(b)\mathcal{A}(b \to a)P(b \to a). \tag{8.8}$$

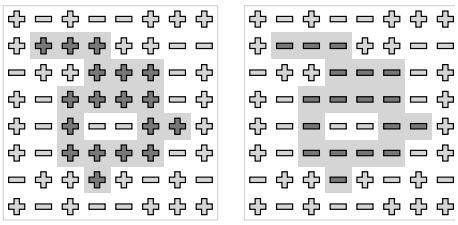

Figure 8.9: Ising-model configurations connected through a cluster flip. In a, 16 edges $\{+,-\}$ and 14 edges $\{+,+\}$ cross the boundary.

We must now compute the a priori probability $\mathcal{A}(a \to b)$, the probability of stopping the cluster construction process at a given stage rather than continuing and including more sites (see the cluster of gray sites in configuration a in Fig. 8.9). $\mathcal{A}(a \to b)$ is given by an interior part (the two neighbors inside the cluster) and the stopping probability at the boundary: each sites on the boundary of the cluster was once a pocket site and the construction came to a halt because none of the possible new edges was included. Precisely, the boundary $\partial\mathcal{C}$ of the cluster (with one spin inside and its neighbor outside) involves two types of edge:

$$\left\{\begin{matrix}\text{cluster in } a \\ \text{in Fig. 8.9}\end{matrix}\right\} : \quad \overbrace{\begin{bmatrix} \text{inside} & \text{outside} & \# \\ + & - & n_1 \\ + & + & n_2 \end{bmatrix}}^{\text{edges across } \partial\mathcal{C}} \qquad E|_{\partial\mathcal{C}} = n_1 - n_2 \tag{8.9}$$

(in the example of Fig. 8.9, $n_1 = 16$ and $n_2 = 14$). The a priori probability is $\mathcal{A}(a \to b) = \mathcal{A}_{\text{in}} \cdot (1-p)^{n_2}$ because there were n_2 opportunities to let the cluster grow and none was taken. To evaluate the Boltzmann weight, we concentrate on the energy across the boundary $\partial\mathcal{C}$, given in eq. (8.9). It follows that $\pi(a) = \pi_{\text{in}}\pi_{\text{out}}\mathrm{e}^{-\beta(n_1-n_2)}$.

We consider the return move from configuration b back to a (see Fig. 8.9 again), and evaluate

the return probability $\mathcal{A}(b \to a)$ and the Boltzmann weight $\pi(b)$. In the cluster for configuration b, the edges across the boundary $\partial\mathcal{C}$ are now

$$\left\{\begin{matrix}\text{cluster in } b \\ \text{in Fig. 8.9}\end{matrix}\right\} : \overbrace{\begin{bmatrix} \text{inside} & \text{outside} & \# \\ - & - & n_1 \\ - & + & n_2 \end{bmatrix}}^{\text{edges across } \partial\mathcal{C}} \quad E|_{\partial\mathcal{C}} = -n_1 + n_2.$$

The cluster construction probability $\mathcal{A}(b \to a)$ contains the same interior part as before, but a new boundary part $\mathcal{A}(b \to a) = \mathcal{A}_{\text{in}} \cdot (1-p)^{n_1}$, because there were n_1 opportunities to let the cluster grow and again none was accepted. By an argument similar to that above, the statistical weight of configuration b is $\pi(b) = \pi_{\text{in}}\pi_{\text{out}}\mathrm{e}^{-\beta(n_2-n_1)}$. The interior and exterior contributions to the Boltzmann weight are the same as for configuration a. All the ingredients of the detailed-balance condition in eq. (8.8) are now known:

$$\mathrm{e}^{-\beta(n_1-n_2)}(1-p)^{n_2}\mathcal{P}(a \to b) = \mathrm{e}^{-\beta(n_2-n_1)}(1-p)^{n_1}\mathcal{P}(b \to a). \tag{8.10}$$

The ratio of acceptance probabilities is

$$\frac{\mathcal{P}(a \to b)}{\mathcal{P}(b \to a)} = \frac{\mathrm{e}^{-\beta(n_2-n_1)}(1-p)^{n_1}}{\mathrm{e}^{-\beta(n_1-n_2)}(1-p)^{n_2}}, \tag{8.11}$$

and the "Metropolis" choice for $\mathcal{P}$ then is

$$\mathcal{P}(a \to b) = \min\left[1, \left(\frac{\mathrm{e}^{-2\beta}}{1-p}\right)^{n_2}\left(\frac{1-p}{\mathrm{e}^{-2\beta}}\right)^{n_1}\right], \tag{8.12}$$

where some terms have been rearranged.

The algorithm is simplest (and at its peak efficiency) at the magic value $p = 1-\mathrm{e}^{-2\beta}$, when the acceptance probability is equal to one: we simply construct a cluster, then flip it, build another one, turns it over .... This is the algorithm of Wolff (1989), which generalizes the original cluster method of Swendsen and Wang (1987). The algorithm is easily implemented with the help of a pocket $\mathcal{P}$ containing the active sites (see Alg. 8.8 (`cluster-ising`)). A Python implementation in two dozen lines is found here.

The cluster algorithm moves through configuration space with breath-taking speed. A typical cluster flip easily involves $\sim 10^3$ spins in a 64×64 Ising model (see Fig. 8.10) and has the system make a giant leap. Running such a insightful code makes us understand the great potential payoff from investments in algorithm design.

8.3.2 Cluster algorithms for hard-sphere systems

Here, we discuss the geometric cluster algorithm.

8.3.3 Dimers on a lattice—cluster algorithms

Here, we discuss the geometric cluster algorithm, as applied to dimers on a square lattice.

References

[1] W. Krauth, *Statistical Mechanics: Algorithms and Computations*. Oxford University Press, 2006.

[2] R. P. Feynman, *Statistical mechanics: a set of lectures*. Frontiers in physics, W. A. Benjamin, Reading, Massachusetts., 1972.

procedure `cluster-ising`
input $\{\sigma_1, \ldots, \sigma_N\}$
$j \leftarrow$ `nran` $(1, N)$
$\mathcal{C} \leftarrow \{j\}$
$\mathcal{P} \leftarrow \{j\}$
while $\mathcal{P} \neq \emptyset$**:**
  $k \leftarrow$ any element of $\mathcal{P}$
  for $(\forall\ l \notin \mathcal{C}$ with l neighbor of $k,\ \sigma_l = \sigma_k)$**:**
    if $(\texttt{ran}(0,1) < p)$ **then**
      $\mathcal{P} \leftarrow \mathcal{P} \cup \{l\}$
      $\mathcal{C} \leftarrow \mathcal{C} \cup \{l\}$
  $\mathcal{P} \leftarrow \mathcal{P} \setminus \{k\}$
for $\forall k \in \mathcal{C}$**:**
  $\sigma_k \leftarrow -\sigma_k$
output $\{\sigma_1, \ldots, \sigma_N\}$

Algorithm 8.8: `cluster-ising`. Cluster algorithm for the Ising model at the magic value $p = 1 - \mathrm{e}^{-2\beta}$.

[3] J. G. Propp and D. B. Wilson, "Exact sampling with coupled Markov chains and applications to statistical mechanics," *Random Structures & Algorithms*, vol. 9, no. 1-2, pp. 223–252, 1996.

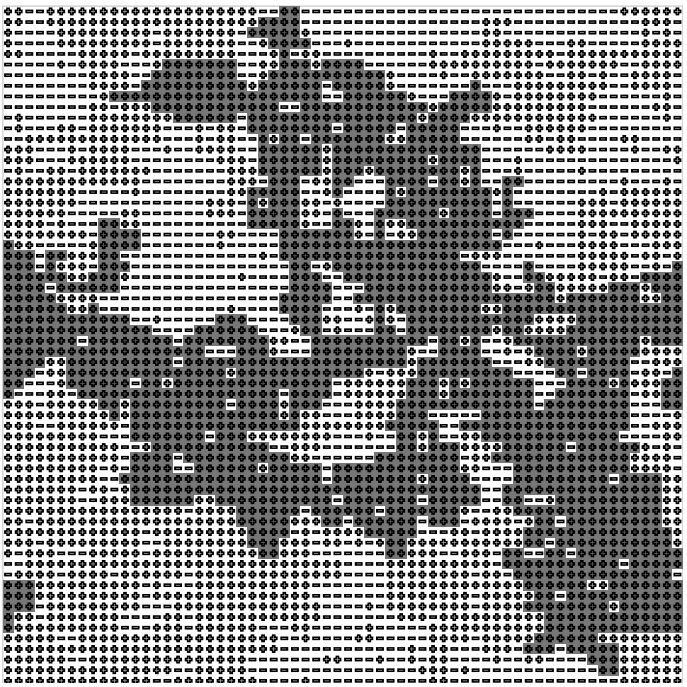

Figure 8.10: Large cluster with 1548 "up" spins in a 64×64 Ising model with periodic boundary conditions (from Alg. 8.8 (`cluster-ising`), $\beta = 0.43$).