# Ising Model

Dieter W. Heermann

Monte Carlo Methods

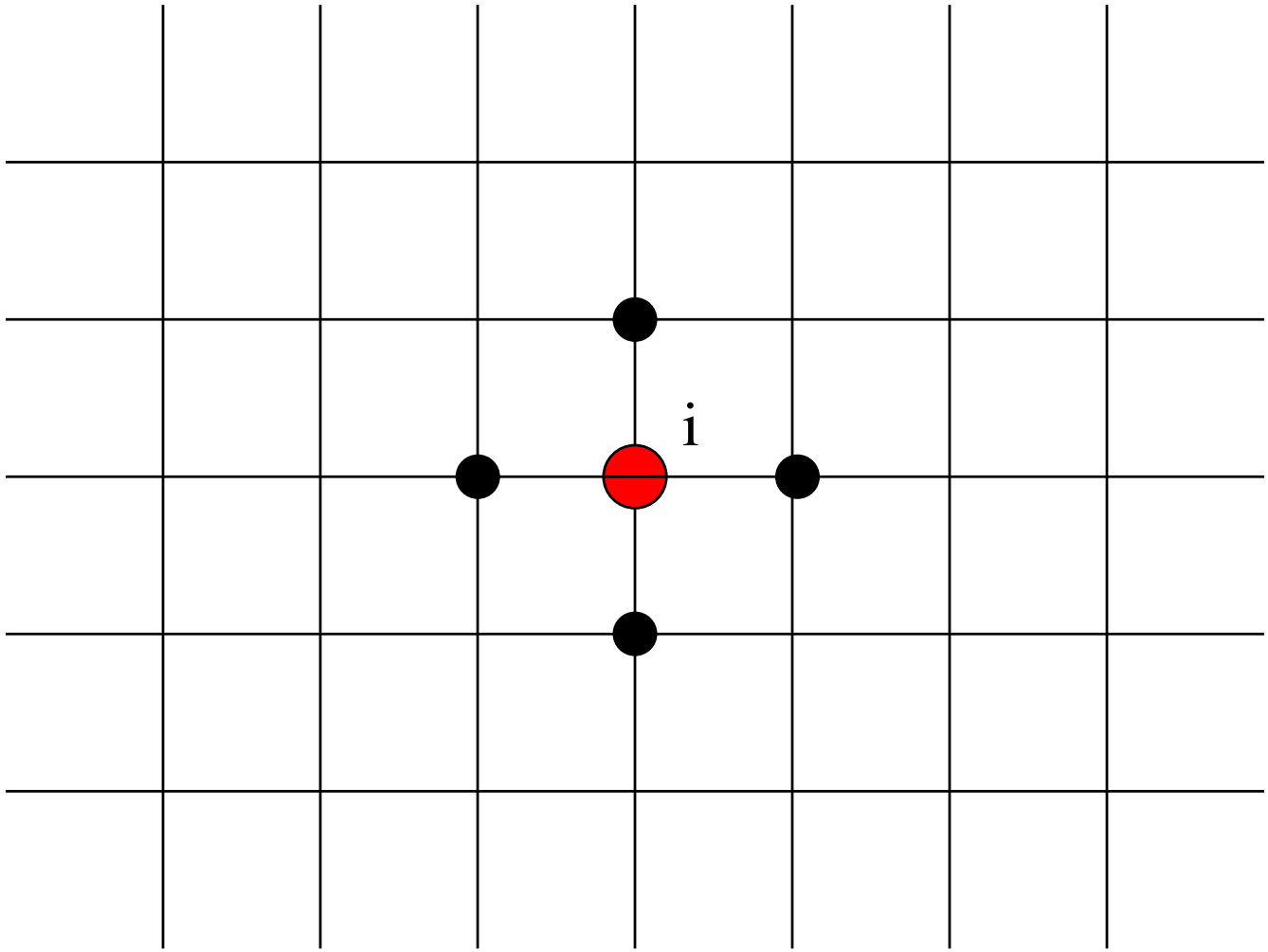
2018

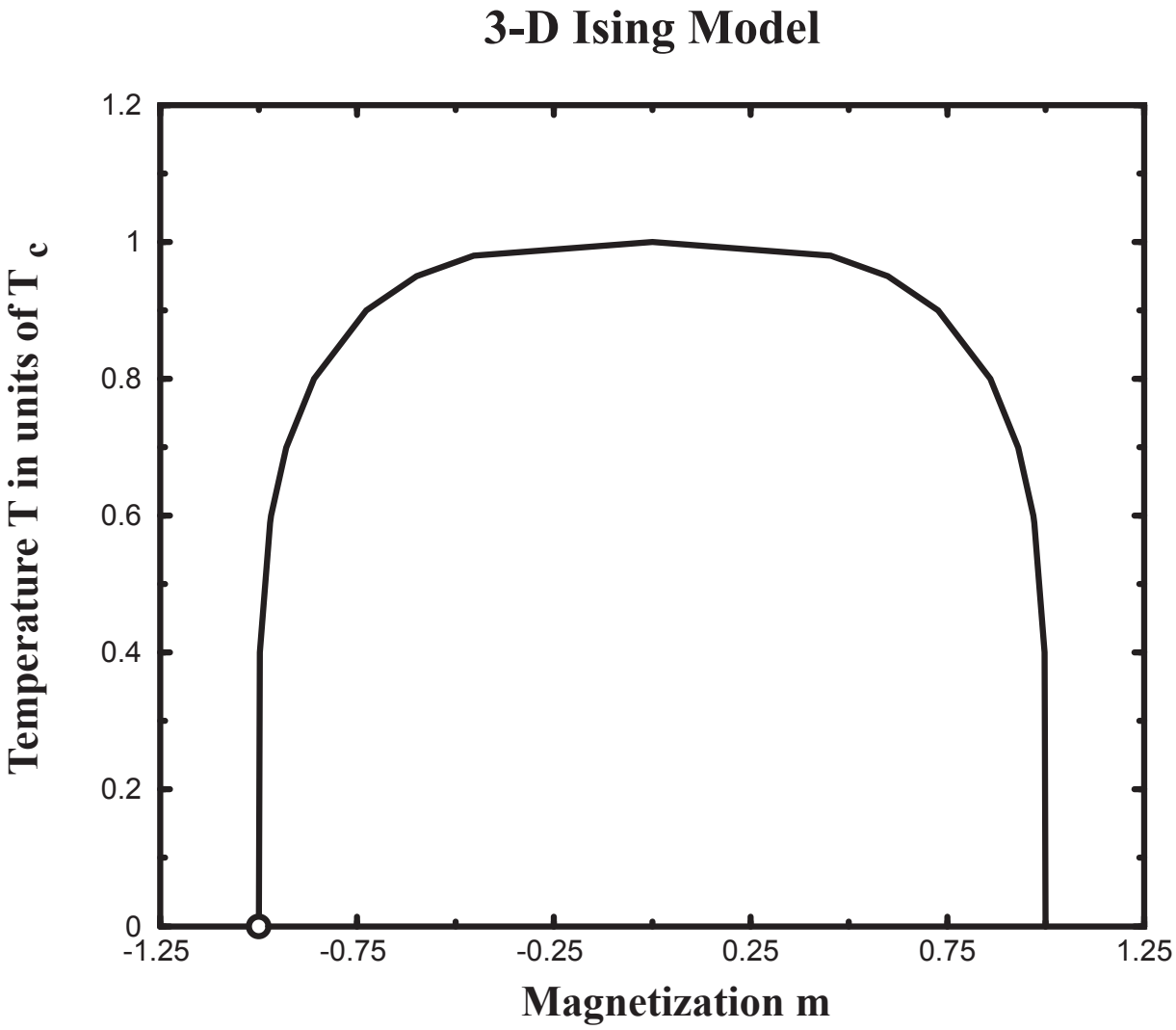
The Ising model [1] is defined as follows:

- Let $G = L^d$ be a d-dimensional lattice.
- Associated with each lattice site $i$ is a spin $s_i$ which can take on the values $+1$ or $-1$.
- The spins interact via an exchange coupling $J$. In addition, we allow for an external field $H$.
- The Hamiltonian reads

$$\mathcal{H} = -J \sum_{\langle i,j \rangle} s_i s_j + \mu H \sum_i s_i \tag{1}$$

- The first sum on the right-hand side of the equation runs over nearest neighbours only.
- The symbol $\mu$ denotes the magnetic moment of a spin. If the exchange constant $J$ is positive, the Hamiltonian is a model for ferromagnetism, i.e., the spins tend to align parallel.
- For $J$ negative the exchange is anti ferromagnetic and the spins tend to align antiparallel. In what follows we assume a ferromagnetic interaction $J > 0$.


i


3-D Ising Model
Temperature T in units of T c
1.2
1
0.8
0.6
0.4
0.2
0
-1.25
-0.75
-0.25
0.25
0.75
1.25
Magnetization m

- Let $E$ be the fixed energy and suppose that a spin configuration $s = (s_1, ..., s_N)$ was constructed with the required energy.
- We set the demon energy to zero and let it travel through the lattice.
- At each site the demon attempts to flip the spin at that site.
- If the spin flip lowers the system energy, then the demon takes up the energy and flips the spin.
- On the other hand, if a flip does not lower the system energy the spin is only flipped if the demon carries enough energy.
- A spin is flipped if

$$E_D - \Delta\mathcal{H} > 0 \tag{2}$$

and the new demon energy is

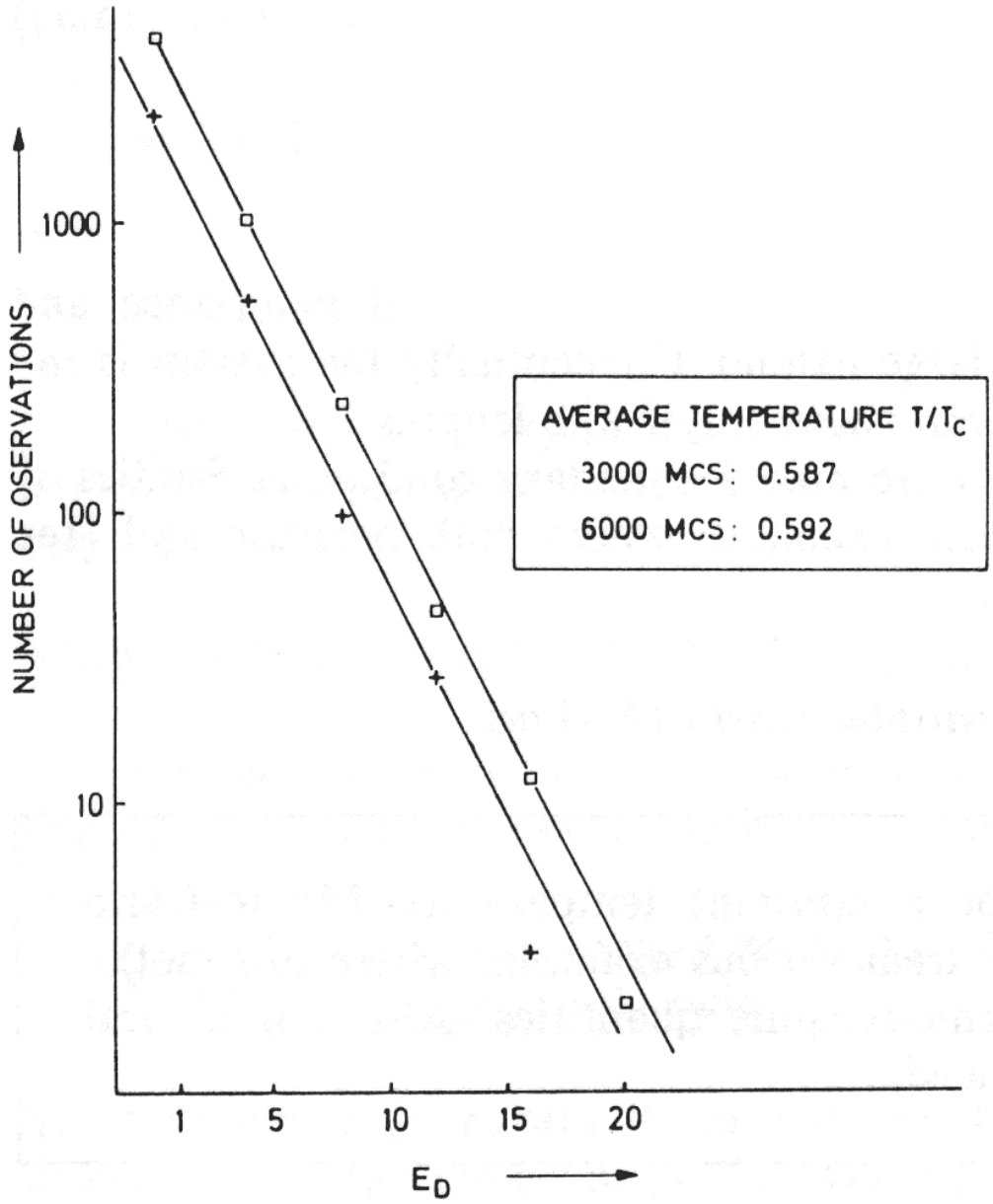
$$E_D = E_D - \Delta\mathcal{H} \tag{3}$$

- After having visited all sites one time unit has elapsed and a new configuration is generated.
- In Monte-Carlo method language the time unit is called the MC step per spin.
- After the system has relaxed to thermal equilibrium, i.e., after $n_0$ Monte-Carlo Steps (MCS), the averaging is started. For example, we might be interested in the magnetization.
- Let $n$ be the total number of MCS, then the approximation for the magnetization is

$$< m >= \frac{1}{n - n_0} \sum_{i \geq n_0}^{n} m(s_i) \tag{4}$$

where $s_i$ is the ith generated spin configuration. Since the demon energies ultimately become Boltzmann distributed, it is easy to show that

$$\frac{J}{k_B T} = \frac{1}{4} \ln \left( 1 + 4 \frac{J}{\langle E_D \rangle} \right) \tag{5}$$


NUMBER OF OSERVATIONS
1000
100
10
1
5
10
15
20
$E_D$
AVERAGE TEMPERATURE $T/T_C$
3000 MCS: 0.587
6000 MCS: 0.592

# Metropolis-Hastings Monte Carlo

- The simplest and most convenient choice for the actual simulation is a transition probability involving only a single spin; all other spins remain fixed.
- It should depend only on the momentary state of the nearest neighbours.
- After all spins have been given the possibility of a flip a new state is created. Symbolically, the *single-spin-flip* transition probability is written as

$$W_i(s_i) : (s_1, ..., s_i, ..., s_N) \longrightarrow (s_1, ..., -s_i, ..., s_N)$$

  where $W_i$ is the probability per unit time that the ith spin changes from $s_i$ to $-s_i$.
- With such a choice the model is called the single-spin-flip Ising model (Glauber).

- Let $P(s)$ be the probability of the state $s$. In thermal equilibrium at the fixed temperature $T$ and field $K$, the probability that the $i$-th spin takes on the value $s_i$ is proportional to the Boltzmann factor

$$P_{eq}(s_i) = \frac{1}{Z} exp\left(\frac{-\mathcal{H}(s_i)}{k_B T}\right)$$

  The fixed spin variables are suppressed.

- We require that the detailed balance condition be fulfilled:

$$W_i(s_i) P_{eq}(s_i) = W_i(-s_i) P_{eq}(-s_i)$$

  or

$$\frac{W_i(s_i)}{W_i(s_i)} = \frac{P_{eq}(-s_i)}{P_{eq}(s_i)}$$

- It follows that

$$\frac{W_i(s_i)}{W_i(s_i)} = \frac{\exp(-s_i/E_i)}{\exp(s_i/E_i)}$$

where

$$E_i = J \sum_{\langle i,j \rangle} s_j$$

- The derived conditions do not uniquely specify the transition probability $W$.
- The *Metropolis function*

$$W_i(s_i) = min\left\{\tau^{-1}, \tau^{-1} exp(-\Delta\mathcal{H}/k_B T)\right\}$$

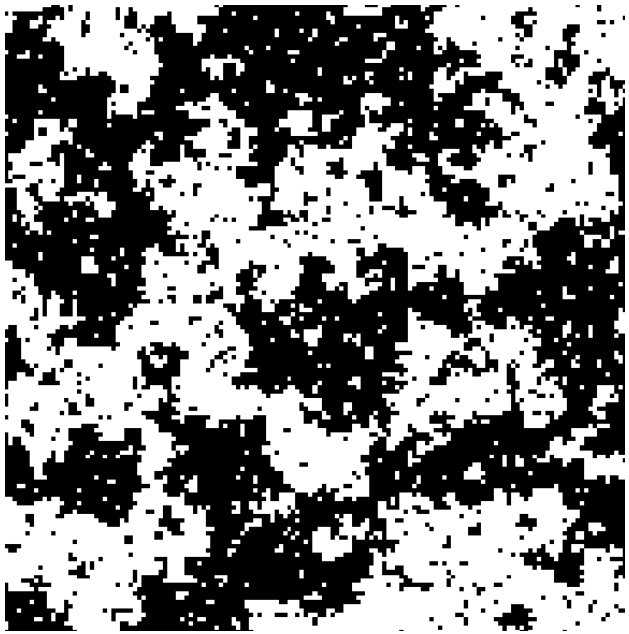
- and the *Glauber function*

$$W_i(s_i) = \frac{(1 - s_i \tanh E_i/k_B T)}{2\tau}$$

where $\tau$ is an arbitrary factor determining the time scale.

Algorithmically the Metropolis MC method looks as follows:

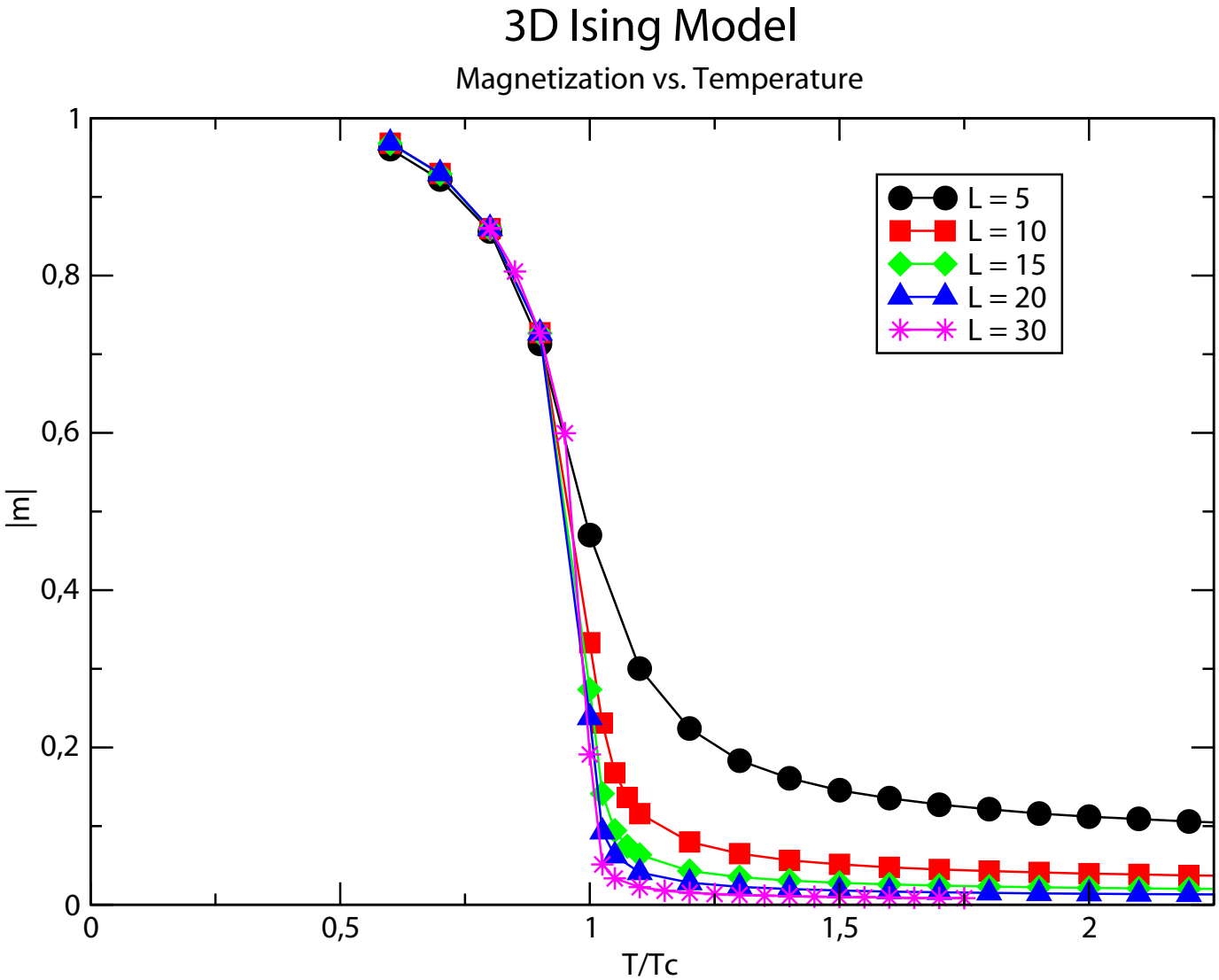
1: Specify an initial configuration.
2: Choose a lattice site $i$.
3: Compute $W_i$.
4: Generate a random number $R \in [0, 1]$.
5: **if** $W_i(s_i) > R$ **then**
6: $\quad s_i \rightarrow -s_i$
7: **else**
8: $\quad$ Otherwise, proceed with Step 2 until *MCSMAX* attempts have been made.
9: **end if**



Java program can be found here

```
#include <iostream.h>
#include <math.h>

# define L  10

int    main(int argc, char *argv[])
{
    int mcs,i,j,k,ip,jp,kp,in,jn,kn;
    int old_spin,new_spin,spin_sum;
    int old_energy,new_energy;
    int mcsmax;
    int spin[L][L][L];
    int seed;
    double r;
    double beta;
    double energy_diff;
    double mag;

 mcsmax = 100;
 beta = 0.12;       // beta = J/kT  KC = 0.2216544 Talapov and Blöte (1996)
 seed = 4711;

 srand(seed);
    for (i=0;i<L;i++) {
        for (j=0;j<L;j++) {
            for (k=0;k<L;k++) {

             spin[i][j][k] = -1;
            }
        }
    }
 mag = - L*L*L;

    // Loop over sweeps
    for (mcs=0;mcs<mcsmax;mcs++) {

        // Loop over all sites
        for (i=0;i<L;i++) {
            for (j=0;j<L;j++) {
                for (k=0;k<L;k++) {

                   // periodic boundary conditions
                 ip = (i+1) % L;
                 jp = (j+1) % L;
                 kp = (k+1) % L;
                 in = (i+L-1) % L;
                 jn = (j+L-1) % L;
                 kn = (k+L-1) % L;

                 old_spin = spin[i][j][k];
                 new_spin = - old_spin;

                   // Sum of neighboring spins
                spin_sum = spin[i][jp][k] + spin[ip][j][k] +
                spin[i][jn][k] + spin[in][j][k] +
                spin[i][j][kn] + spin[i][j][kp];

                old_energy = - old_spin * spin_sum;
                new_energy = - new_spin * spin_sum;
                energy_diff = beta * (new_energy - old_energy);
```

The C program can be found here


3D Ising Model
Magnetization vs. Temperature
|m|
T/Tc
L = 5
L = 10
L = 15
L = 20
L = 30
0
0,2
0,4
0,6
0,8
1
0
0,5
1
1,5
2

# Global Algorithms

See lecture on cluster and multi-grid algorithms

 E. Ising: Z. Phys. 31,253 (1925)

 M. Creutz: Phys. Rev. Lett. 50, 1411 (1983)