Zastosowanie metod Monte Carlo: Symulacje fizyki statystycznej - model Isinga

Agnieszka Mach, Robert Wojtaszek

September 2020

Spis treści

1 Metoda Monte Carlo

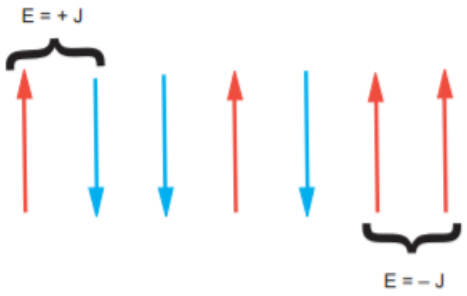
Jest to metoda stosowana do modelowania matematycznego procesów zbyt złożonych (takich jak np. obliczanie całek, bądź łańcuchów procesów statstycznych), aby można było przewidzieć ich wynik za pomocą podejścia analitycznego. Co więcej, niezwykle istotną rolę w tej metodzie odgrywa wybór przypadkowy (i.e. losowanie) wielkości, które charakteryzują proces, który musi być znany.
Metoda Monte Carlo to jedna z najczęściej używanych metod symulacyjnych. Jedną z najczęstszych aplikacji ów metody jest całkowanie numeryczne.
Jej zaletami jest zdecydowanie możliwość rozwiązywania trudnych i złożonych problemów przy jednoczesnej prostej formie zastąpienia rozwiązań analitycznych, dzięki czemu użytkownik "wolny jestód skomplikowanych teorii i wzorów.
Wadliwość metody Monte Carlo polega na otrzymanym wyniku będącym raptem przybliżeniem, który zależy od jakości generatora liczb pseudolosowych.
Typowym przykładem zastosowania ów metody jest modelowanie wyniku zderzenia cząstki o wysokiej energii z jądrem złożonym (z zaznaczeniem, że każdy

akt zderzenia elementarnego modelowany jest oddzielnie poprzez losowanie liczby, rodzaju, kąta emisji, energii itp. cząstek wtórnych emitowanych w wyniku takiego zderzenia). W tym celu modeluje się los każdej z cząstek wtórnych. Kontynuując taką procedurę, można otrzymać pełny opis „sztucznie generowanego" procesu złożonego. Po zebraniu dostatecznie dużej liczby takich informacji można zestawić ich charakterystyki z obserwowanymi wynikami doświadczalnymi, potwierdzając lub negując słuszność poczynionych w całej procedurze założeń.

2 Model Isinga

Model ten jest modelem matematycznym wykorzystywanym w mechanice statystycznej do badań nad przejściami fazowymi. Opisuje go układ dyskretnych zmiennych **S** (spinów), które przyjmują wartości +1 bądź też -1 zlokalizowane na każdym węźle sieci. Energia oddziaływania pomiędzy parą spinów przyjmuje więc jedną z dwóch wartości, która zależna jest od ich wzajemnej orientacji (czy ułożone one są zgodnie ze sobą czy też przeciwnie).

Formalnie model Isinga jest modelem **ferromagnetyka** na danej sieci. Wiadomo, że ferromagnetyk poniżej temperatury Curie wykazuje sponitaniczną magnetyzację (zanika ona powyżej temperatury Curie). Substancja ta staje się wówczas paramagnetykiem.



Rysunek 1: Sieć 1-D spinów N użyta w modelu magnetyzmu Isinga. Energia interakcji między parami najbliższych sąsiadów E = ± J jest pokazana dla spinów wyrównanych i przeciwnych.

Jako nasz model rozważamy N dipoli magnetycznych umocowanych na ogniwach łańcucha liniowego (Rysunek 1.). (Prostym uogólnieniem jest obsługa sieci dwuwymiarowych i trójwymiarowych). Ponieważ cząstki są stałe, ich pozycje i pędy nie są zmiennymi dynamicznymi, więc zajmujemy się jedynie kwestią spinów. Zakładamy, że cząstka w miejscu i ma spin si, czyli albo góra czy dół:

$$s_i \equiv s_{z,i} = \pm\frac{1}{2}.$$

Każda konfiguracja cząstek N jest opisana przez wektor stanu kwantowego

$$|\alpha_j\rangle = |s_1, s_2, \ldots, s_N\rangle = \left\{\pm\frac{1}{2},\ \pm\frac{1}{2},\ \ldots\right\}, \quad j = 1, \ldots, 2^N.$$

Ponieważ spin każdej cząstki może przyjąć jedną z dwóch wartości jest ich więc 2N różnych możliwych stanów cząstek N w układzie. Ponieważ stałych cząstek nie można zamieniać to nie musimy zajmować się symetrią funkcji falowej.
Energia powstaje w wyniku interakcji spinów z sobą samym oraz z zewnętrznym polem magnetycznym B. Mechanika kwantowa mówi nam, że spin elektronu i moment magnetyczny są do siebie proporcjonalne, więc zachodzi wzajemne oddziaływanie magnetyczne dipol-dipol, będące odpowiednikiem interakcji spin-spin. Zakładamy, że każdy dipol oddziałuje z zewnętrznym pole magnetycznym oraz z najbliższym sąsiadem poprzez potencjał:

$$V_i = -J\mathbf{s}_i \cdot \mathbf{s}_{i+1} - g\mu_b\, \mathbf{s}_i \cdot \mathbf{B}.$$

Stała J nazywana jest energią wymiany i jest miarą siły spin–interakcja spinowa. Stała g jest stosunkiem żyromagnetycznym, czyli stałą proporcjonalności między momentem pędu cząstki a momentem magnetycznym. Stała μb = e $\hbar$ / ($2m_e$c) to magneton Bohra (podstawowa miara momentów magnetycznych).
Energia układu w stanie ω k jest wartością oczekiwaną sumy potencjału V ponad spinami cząstek:

$$E_{\alpha_k} = \left\langle \alpha_k \middle| \sum_i V_i \middle| \alpha_k \right\rangle = -J\sum_{i=1}^{N-1} s_i s_{i+1} - B\mu_b \sum_{i=1}^{N} s_i.$$

Pozorny paradoks w modelu Isinga pojawia się, gdy wyłączamy zewnętrzne pole magnetyczne, tym samym eliminując preferowany kierunek w przestrzeni. Oznacza to, że średnie namagnesowanie powinno zniknąć, mimo że najniższy stan energii miałyby wszystkie spiny w jednej linii. Wynika to z tego, że B = 0 jest niestabilne. Nawet jeśli wszystkie spiny są wyrównane nie istnieje nic co mogłoby powstrzymać ich spontaniczne odwrócenie. Faktycznie, naturalne materiały magnetyczne mają wiele domen skończonych ze wszystkimi spinami wyrównanymi, ale z domenami wskazującymi na różne kierunki. Niestabilności w domenach zmienia kierunek (dla uproszczenia załóżmy, że B = 0 co oznacza, że spiny oddziałują ze sobą). W tym momencie należy pamiętać o tym, ze nie istnieje preferowany kierunek w przestrzeni (np. może być konieczne przyjęcie

wartości bezwzględnej całkowitego spinu przy obliczaniu namagnesowania).

Równowaga wyrównania spinów zależy od znaku wymiany energii J.

- Jeśli J¿ 0 to najniższy stan energii będzie miał tendencję do ustawiania sąsiednich spinów.
- Jeśli temperatura jest wystarczająco niska, to stan podstawowy będzie ferromagnesem z ustawionymi wszystkimi spinami
- Jeśli J ¡0 to najniższy stan energii będzie miał tendencję do posiadania sąsiadów z przeciwnymi spinami.
- Jeśli temperatura jest dostatecznie niska, to stan podstawowy będzie antyferromagnesem ze zmiennymi spinami.

Prosty model Isinga 1-D ma swoje limity. Chociaż jest on dokładny w opisywaniu układu w równowadze termicznej, to nie jest on dokładny w opisywaniu podejścia do ównowagi termicznej.

Interesującym aspektem materiałów magnetycznych jest istnienie temperatury krytycznej, tzw. temperatury Curie, powyżej której ogólne namagnesowanie zasadniczo zanika. Poniżej temperatury Curie natomiast stan kwantowy materiału ma porządek dalekiego zasięgu wykraczający poza makroskopowy wymiary; powyżej temperatury Curie istnieje tylko porządek krótkiego zasięgu wymiaru wykraczający poza atom. Mimo że model 1-D Isinga przewiduje realistyczne zależności temperaturowe dla wielkości termodynamicznej, tomodel jest zbyt prosty, aby wspierać przemianę fazową. W tym celu stosuje się więc modele 2-D i 3-D Isinga wspierające przejścia fazowe temperatury Curie.

3 Model Isinga w jednym wymiarze

W układzie jednowymiarowym można nałożyć periodyczne warunki brzegowe S _$N + 1$ = S _1
Hamiltonian dla takiego układu wygląda następująco:

$$H = -J\sum_i S_i S_{i+1} - h\sum_i S_i = -J\sum_i S_i S_{i+1} - \frac{1}{2}h\sum_i S_i - \frac{1}{2}h\sum_i S_{i+1} = -\sum_i \left(JS_i S_{i+1} + \frac{1}{2}h(S_i + S_{i+1}) \right)$$
$$= -\left(Js_1 S_2 + \frac{1}{2}h(S_1 + S_2) + Js_2 S_3 + \frac{1}{2}h(S_2 + S_3) + Js_3 S_4 + \frac{1}{2}h(S_3 + S_4) + \ldots + Js_N S_1 + \frac{1}{2}h(S_N + S_1) \right).$$

Statystyczna suma stanów:

$$
\begin{aligned}
Z &= \sum_{S_1,\ldots,S_N} \exp(-\beta H) = \sum_{S_1,\ldots,S_N} \exp\left[\beta \sum_i \left(J S_i S_{i+1} + \frac{1}{2} h(S_i + S_{i+1}) \right)\right] \\
&= \sum_{S_1,\ldots,S_N} \exp\left[\beta \left(J s_1 S_2 + \frac{1}{2} h(S_1 + S_2) + J s_2 S_3 + \frac{1}{2} h(S_2 + S_3) + J s_3 S_4 + \frac{1}{2} h(S_3 + S_4) + \ldots + J s_N S_1 + \frac{1}{2} h(S_N + S_1) \right)\right] \\
&= \sum_{S_1,\ldots,S_N} \exp\left[\beta \left(J s_1 S_2 + \frac{1}{2} h(S_1 + S_2) \right)\right] \exp\left[\beta \left(J s_2 S_3 + \frac{1}{2} h(S_2 + S_3) \right)\right] \exp\left[\beta \left(J s_3 S_4 + \frac{1}{2} h(S_3 + S_4) \right)\right] \ldots \exp\left[\beta \left(J s_N S_1 + \frac{1}{2} h(S_N + S_1) \right)\right] \\
&= \sum_{S_1,\ldots,S_N} M_{S_1,S_2} M_{S_2,S_3} \ldots M_{S_N,S_1} = (*),
\end{aligned}
$$

gdzie:

$$
M_{S_1,S_2} = M_{S_2,S_3} = \ldots = M_{S_N,S_1} = M_{S_i,S_{i+1}} = M = \exp\left[\beta \left(J s_i S_{i+1} + \frac{1}{2} h(S_i + S_{i+1}) \right)\right].
$$

Możliwe sa cztery „warianty” M:

$$
\begin{array}{c|cc}
 & s_i = -1 & s_i = +1 \\
\hline
s_{i+1} = -1 & e^{\beta(J-h)} & e^{-\beta J} \\
s_{i+1} = +1 & e^{-\beta J} & e^{\beta(J+h)}
\end{array}.
$$

Wracając więc do sumy statystycznej

$$
Z = (*) = Tr(M^N) = Tr(M \cdot M \cdot M \cdot \ldots \cdot M) = (**).
$$

Macierz M można przedstawić w postaci

$$
M = U^\dagger M^D U \text{ gdzie } M^D
$$

gdzie M $\hat{D}$ jest macierzą diagonalną, a

$$
UU^\dagger = 1
$$

$$
Z = (**) = Tr(U^\dagger M^D U U^\dagger M^D U U^\dagger \ldots M^D U) = Tr(U^\dagger (M^D)^N U) = (***).
$$

M$\hat{D}$ jest macierzą diagonalną, jej postać jest więc następująca:

$$
M^D = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}.
$$

Natomiast

$$
(M^D)^N = \begin{pmatrix} {\lambda_1}^N & 0 \\ 0 & {\lambda_2}^N \end{pmatrix}.
$$

Wyznaczenie wartości własnych dla M:

$$
\begin{aligned}
\det M &= \det \begin{pmatrix} \exp(\beta J + \beta h) - \lambda & \exp(-\beta J) \\ \exp(-\beta J) & \exp(\beta J - \beta h) - \lambda \end{pmatrix} \\
&= (\exp(\beta J + \beta h) - \lambda)(\exp(\beta J - \beta h) - \lambda) - \exp(2\beta J) \\
&= 2\sinh(2\beta J) - \lambda 2\exp(\beta J)\cosh(\beta h) + \lambda^2, \\
\lambda_1 &= \exp(\beta J)\left[\cosh(\beta h) + \sqrt{\cosh^2(\beta h) - 2\exp(-2\beta J)\sinh(2\beta J)}\right], \\
\lambda_2 &= \exp(\beta J)\left[\cosh(\beta h) - \sqrt{\cosh^2(\beta h) - 2\exp(-2\beta J)\sinh(2\beta J)}\right].
\end{aligned}
$$

Wybierając największą wartość własną macierzy:

$$\lambda_1 > \lambda_2,$$

otrzymujemy że suma statystyczna jest równa:

$$Z = (***) = {\lambda_1}^N + {\lambda_2}^N = {\lambda_1}^N \cdot \left(1 + \left(\frac{\lambda_2}{\lambda_1}\right)^N\right).$$

Jeśli

$$\lambda_2 < \lambda_1 \text{ to: } \left(\frac{\lambda_2}{\lambda_1}\right)^N \ll 1.$$

$$Z = {\lambda_1}^N.$$

Faza stabilna jest określona przez największą wartość własną. Przejście fazowe (np. między fazą ferro i paramagnetyczną) zachodzi wtedy, gdy zrównują się wartości własne.

Namagnesowanie w takim wypadku jest równe:

$$
\begin{aligned}
m &= \frac{1}{N} kY \frac{\partial}{\partial h} \ln Z \\
&= \frac{1}{\beta}\frac{\partial}{\partial h} \ln \exp(\beta h) \cdot \left[\cosh(\beta h) + \sqrt{\cosh^2(\beta h) - 2\exp(-2\beta J)\sinh(2\beta J)}\right] \\
&= \frac{1}{\beta}\frac{1}{\lambda_1}\exp(\beta J) \cdot \left[\beta\sinh(\beta h) + \frac{2\beta\sinh(\beta h)\cosh(\beta h)}{2\sqrt{\cosh^2(\beta h) - 2\exp(-2\beta J)\sinh(2\beta J)}}\right] \\
&= \frac{\exp(\beta J)\sinh(\beta h)}{\lambda_1} \cdot \left[\frac{\sqrt{\cosh^2(\beta h) - 2\exp(-2\beta J)\sinh(2\beta J)} + \cosh(\beta h)}{\sqrt{\cosh^2(\beta h) - 2\exp(-2\beta J)\sinh(2\beta J)}}\right].
\end{aligned}
$$

Czyli ostatecznie namagnesowanie:

$$m = \frac{\sinh(\beta h)}{\sqrt{\cosh^2(\beta h) - 2\exp(-2\beta J)\sinh(2\beta J)}}.$$

4 Zastosowanie modelu

```cpp
//bladzenie losowe
//kompilacja: g++  program.cpp -lgsl -lgslcblas

    #include <stdio.h>
    #include <math.h>
    #include <gsl/gsl_rng.h>
    #include <gsl/gsl_randist.h>

    #include <iostream>
    #include <fstream>
    #include <string>
    using namespace std;


//////////////deklaracja klasy////////////////////////////////////
class Ising
{
public:

   Ising( int N, double T, double J, double H );                  //konstruktor
   ~Ising();                                    //destruktor
   void makeStep( void );                              //wyknaj jeden krok i kumuluje statystyki
   void saveToFile( string filename );                    //zapisz konfiguracje do pliku
   double calculateE( void );                          //oblicza energie ukladu
   double getM( void );                                //oblicza srednia magnetyzacje ukladu


private:
   Ising(void);    //zablokowanie tworzenia obiektow konstruktorem domyslnym
   double calculateM( void ); //oblicza magnetyzacje ukladu
   void makeMCStep( void );   //wykonuje krok Monte Carlo

   int N;          //liczba spinow
   int * spiny;        //wskaznik do tablicy ze spinami
   int kroki;      //liczbna krokow symulacji
   int Nterm;      //liczba krokow termalizacji
   gsl_rng * rand;     //generator liczb losowych
```

```
    double kB;      //stala Boltzmanna
    double E;       //energia danej konfiguracji
    double T;       //temperatura ukladu utrzymywana przez termostat
    double J;       //stala sprzezenia pomiedzy spinami
    double H;       //zewnetrzne pole magnetyczna
    double tempM;       //zmienna pomocnicza do pamietania magnetyzacji


};

 /////////implementacja metod klasy//////////////////////////////

Ising :: Ising( int N, double T, double J, double H )
 : N(N), kroki(0), Nterm(100), kB(1.0), T(T), J(J), H(H)
{
    //inicjalizacja generatora liczb losowych
    const gsl_rng_type * randType;
    gsl_rng_env_setup();
    randType = gsl_rng_default;
    rand = gsl_rng_alloc ( randType );


    //tworzenie tablicy do przechowywania pozycji pionkow
    spiny = new int [N];
```

```

    double p;   //pomocnicza zmienna do pamietania wygenerowanej liczby
    for( int i = 0; i < N; i++)
    {
        p=(0.5 - gsl_rng_uniform_pos(rand));
        //p= - gsl_rng_uniform_pos(rand);
        if( p > 0.0)
        {
          spiny[i] = 1;
        }
        else
          spiny[i] = -1;

    }

    //oblicz energie poczatkowa
    E = calculateE();


    //wykonaj termalizacje
    for( int i =0; i < Nterm; i++ )
        makeMCStep();

};


```

```
89
90  Ising::~Ising()
91  {
92
93      delete [] spiny;
94
95      spiny = 0;
96
97
98      //usuniecie generatora
99         gsl_rng_free (rand);
100
101 };
102
103 void Ising :: saveToFile( string filename )
104 {
105     ofstream plik;
106     plik.open(filename.c_str());
107
108     for ( int i = 0; i < N; i++)
109             plik << i << "\t" << spiny[i]  << endl;
110
111     plik.close();
112
113     return;
114 };
115
```

```
116
117 double Ising::calculateE(void )
118 {
119
120     double tempE = 0.0;     //zmienna pomocnicza do obliczania energii
121     double tempKorelacje = 0.0; //korelacje pomiedzy spinami
122     double tempSpinSum = 0.0;   //suma spinow
123
124     //srodek
125     for(  int i = 1; i < N - 2; i++ )
126     {
127         tempKorelacje += spiny[i] * spiny[i+1];
128         tempSpinSum += spiny[i];
129     }
130     //brzegi - periodyczne warunki brzegowe
131     tempKorelacje += spiny[N-1] * spiny[0];
132     tempSpinSum += spiny[0];
133     tempKorelacje += spiny[0] * spiny[N-1];
134     tempSpinSum += spiny[N];
135
136     tempE = -J * tempKorelacje - H * tempSpinSum;
137
138     return ( tempE );
139 }
140
141 double Ising::calculateM(void )
142 {
143     double tempSpinSum = 0.0;   //suma spinow
144
145     for(  int i = 0; i < N; i++ )
146     {
147         tempSpinSum += spiny[i];
148     }
149
150     return( tempSpinSum / double( N ) );
```

```
151     }
152
153     double Ising::getM(void )
154     {
155         return( tempM / double ( kroki - Nterm ) );
156     }
157
158     void Ising::makeStep(void )
159     {
160         makeMCStep();
161         tempM += calculateM();
162
163         return;
164     }
165
166
167     void Ising::makeMCStep( void )
168     {
169
170         int wezel = 0;      //wybrany wezel
171         double newE = 0.0;  //nowa energia ukladu
172
173     //kork Monte Carlo
174     for ( int i = 0; i < N + 1; i++)
175     {
176         //algorytm Metropolisa:
177
178         //wylosuj numer wezla do zmiany
179         wezel = gsl_rng_uniform_int ( rand, N );
180
181         //odwroc spin w tym wezle (spin flip)
182         spiny[wezel] = - spiny[wezel];
183
184         //oblicz nowa energie ukladu
185         newE = calculateE();
186
187         //jezeli energia ukladu sie zmniejszyla to akceptuj nowa konfiguracje
188         if ( newE < E )
```

```
189         {
190         E = newE;
191         kroki++;
192 
193         continue;
194         }
195         //w przeciwnym wypadku akceptuj warunkowo:
196         else
197         {
198         //losuj liczbe z przedzialu jednostkowego
199         double p =  gsl_rng_uniform_pos(rand);
200 
201         //akceptuj warunkowo konfiguracje
202         if ( p < exp(- ( newE - E ) / ( kB * T ) ) )
203         {
204             E = newE;
205             kroki++;
206 
207             continue;
208         }
209 
210         //w przeciwnym wypadku wroc do starej konfiguracji
211         spiny[wezel] = - spiny[wezel];
212         kroki++;
213 
214         }
215 
216     }
217         return;
218     };
219 
220     //////////main////////////////////////////////////////
221 
222 
```

```
223
224     int main (void)
225     {
226
227
228       double T = 0.1;
229
230
231     ofstream plik;
232     plik.open( "magnetyzacja.txt" );
233
234
235
236   //petla po temperaturze
237   for (int i = 0; i < 100; i++)
238   {
239   Ising * Is = new  Ising( 500, T, 0.5, 0.0 );
240
241
242   //petla po krokach - termalizacja + symulacja
243       for (int i = 0; i < 100; i++)
244       {
245       Is->makeStep();
246       }
247
248       plik << T << "\t" << Is->getM() << endl;
249
250       cout << "Wykonano krok dla T = " << T << endl;
251
252     T += 0.1;
253
254     delete Is;
255
256
257   }
```

```
258
259
260   plik.close();
261
262
263       return 0;
264     }
265
266
267
```

5 Mechanika Statystyczna

Mechanika Statystyczna - zajmuje się równowagą sił i momentów sił działających na nieruchome ciała materialne - czyli układami statycznie zrównoważonymi. Statyka oprócz równowagi ciał stałych zajmuje się także równowagą cieczy oraz gazów (hydrostatyka, aerostatyka). W niektórych symulacjach, takich jak dynamika molekularna problem występuję, bo energia układu jest ustalona. Stany takie są opisywane za pomocą tzw. Zespołu mikokanonicznego. Natomiast w przypadku symulacji termodynamicznych,

które badamy w tym rozdziale, temperatura, objętość, a liczba cząstek pozostaje stała, więc mamy coś, co nazywamy zespołem kanonicznym. Kiedy mówimy, że obiekt ma temperaturę T, mamy na myśli atomy tego obiektu w równowadze termodynamicznej w temperaturze T takiej, że każdy atom ma średnią energię proporcjonalną do T. Chociaż może to być stan równowagi, jest to stan dynamiczny, w którym energia obiektu zmienia się, gdy wymienia on energię z otoczeniem. Rzeczywiście, jednym z najbardziej pouczających aspektów symulacji, którą opracujemy, jest jej wizualizacja ciągłej i przypadkowej wymiany energii, która zachodzi w stanie równowagi. Energia E stanu a w zespole kanonicznym nie jest stała, ale jest rozprowadzana z prawdopodobieństwem P(a) podane przez rozdkład Boltzmana:

$$\mathcal{P}(E_{\alpha_j}, T) = \frac{e^{-E_{\alpha_j}/k_B T}}{Z(T)}, \quad Z(T) = \sum_{\alpha_j} e^{-E_{\alpha_j}/k_B T}$$

k - stała Boltzmana, T - temperatura, W przypadku bardzo dużej liczby cząstek właściwości termodynamiczne modelu 1-D mogą zostać rozwiązane w sposób analityczny

$$U = \langle E \rangle$$

$$\frac{U}{J} = -N \tanh \frac{J}{k_B T} = -N \frac{e^{J/k_B T} - e^{-J/k_B T}}{e^{J/k_B T} + e^{-J/k_B T}} = \begin{cases} N, & k_B T \to 0, \\ 0, & k_B T \to \infty. \end{cases}$$

Również możemy przedstawić ciepło właściwe cząstki i namagnesowania

$$C(k_B T) = \frac{1}{N} \frac{dU}{dT} = \frac{(J/k_B T)^2}{\cosh^2(J/k_B T)}$$

$$M(k_B T) = \frac{N e^{J/k_B T} \sinh(B/k_B T)}{\sqrt{e^{2J/k_B T} \sinh^2(B/k_B T) + e^{-2J/k_B T}}}.$$

W przypadku 2-D Ising model podsiada analityczne rozwiązanie, ale nie jest ono takie proste.

$$\mathcal{M}(T) = \begin{cases} 0, & T > T_c \\ \frac{(1+z^2)^{1/4}(1-6z^2+z^4)^{1/8}}{\sqrt{1-z^2}}, & T < T_c, \end{cases}$$

$$kT_c \simeq 2.269185J, \quad z = e^{-2J/k_B T},$$

gdzie temperatura jest mierzona w jednostkach temperatury Curie Tc.

6 Bibliografia

- Materiały dydaktyczne udostępnione przez prowadzącego: MonteCarlo2.pdf
- http://coin.wne.uw.edu.pl/ tmostowski/pliki/matlab/matlab7.pdf

- https://pl.wikipedia.org/wiki/MetodaMonteCarlo
- http://th-www.if.uj.edu.pl/zfs/gora/statystyczna15/wyklad12.pdf