Das Ising-Modell mit Monte-Carlo und Metropolis

Christian Darsow-Fromm Maximilian Menzel

8. Februar 2013

Inhaltsverzeichnis

1 Das Ising-Modell **2**

2 Die Monte-Carlo-Methode **3**

3 Die Implementierung **4**
3.1 Die Architektur 4
3.2 Simulationsstart 4
3.3 Implementation des Algorithmus' 5
3.4 Die Metropolis-Strategie 5
3.5 Spin-Array 6

4 Simulations-Ergebnisse **7**
4.1 Der kritische Punkt 7
4.2 Hysterese beim Ferromagneten 7
4.3 Domänenwände beim Antiferromagneten 8

1 Das Ising-Modell

Das Ising-Modell ist eine Vereinfachung des Heisenberg-Modells. Die Spins werden nicht als Vektor behandelt, sondern auf $s_i^z = \pm 1$ reduziert.

$$\hat{\mathcal{H}} = -\frac{1}{2} \sum_{ij} J_{ij} S_i^z S_j^z - B_z \sum_{i=1}^{N} S_i^z \tag{1}$$

Deshalb ist das Ising-Modell vor allem dann eine gute Näherung des Heisenberg-Modells, wenn sich durch Anisotropien eine vorgezogene Richtung ergibt, oder wenn sich die Spins an einem Isotropen äußeren Magnetfeld ausrichten. Um das Modell weiter zu vereinfachen, wird in der Regel für J_{ij} eine nächste-Nachbar-Wechselwirkung angenommen. J ist konstant für das gesamte System und es wird nur die Wechselwirkung mit den direkten Nachbarn berechnet.

Mit dem Modell kann der Phasenübergang eines Ferromagneten in zwei und mehr Dimensionen beschrieben werden.

2 Die Monte-Carlo-Methode

Die Monte-Carlo-Methode ermöglicht das Simulieren von Systemen, die auf Wahrscheinlichkeiten beruhen. Mit ihrer Hilfe ist es nicht nötig, komplexe stochastische Formeln zu entwickeln, sondern es reicht ein Zufallszahlengenerator aus. Durch sehr häufige Wiederholung der Messung lassen sich gute statistische Näherungen simulieren. Statistische Messungenauigkeiten dieser Simulationsmethode lassen sich auch bei realen Messungen beobachten. Daher ist Monte-Carlo relativ realitätsnah.

Monte-Carlo und Ising Die Vereinigung von Monte-Carlo mit dem Ising-Modell funktioniert nach folgendem Schema:

1. Wähle zufällig irgendeinen Spin S_i aus und versuche ihn umzudrehen.

2. Berechne die Energiedifferenz

$$\Delta E = E(-S_i) - E(S_i) \tag{2}$$

   Wenn ΔE negativ ist, wird der Spinflip akzeptiert. Ansonsten wird die Entscheidung Metropolis überlassen.
   Metropolis: Wähle eine Zufallszahl r zwischen 0 und 1. Wenn

$$r < e^{-\beta \Delta E} \tag{3}$$

   wird der Flip akzeptiert. $\beta = \frac{1}{k_B T}$ also der Kehrwert der Temperatur T, da bei uns die Boltzmann-Konstante k_B gleich eins ist.

3. Speichere die neue Wahrscheinlichkeit der Spinausrichtung

$$\langle S_i \rangle_{\text{neu}} = \frac{\langle S_i \rangle \cdot N_i + S_i}{N_i + 1}, \tag{4}$$

   wobei N_i die Anzahl der bisherigen Messungen des Spins S_i ist.

4. Die Magnetisierung M ist die Summe der Erwartungswerte der einzelnen Spins S_i, normalisiert mit der Anzahl der Spins N:

$$M = \frac{1}{N} \sum_i \langle S_i \rangle \tag{5}$$

Die Punkte eins bis drei werden immer wiederholt und zum Schluss die Magnetisierung gemessen.

3 Die Implementierung

3.1 Die Architektur

Unser Code ist lässt sich ein primär 3 Bestandteile zerlegen: *Ising.cpp* dient als Einstiegspunkt, *IsingMetropolis.cpp* definiert den Algorithmus und *MetropolisStrategy.h* beschreibt die Struktur der Probe.

Die drei Teile sind durch das Strategie-Pattern, wie in Abbildung 1 dargestellt, lose gekoppelt. Beim Strategie-Pattern wird ein (abstrakter) Algorithmus definiert und die Strategie als Parameter übergeben. Im Algorithmus werden dann Methoden der Strategie ausgeführt.

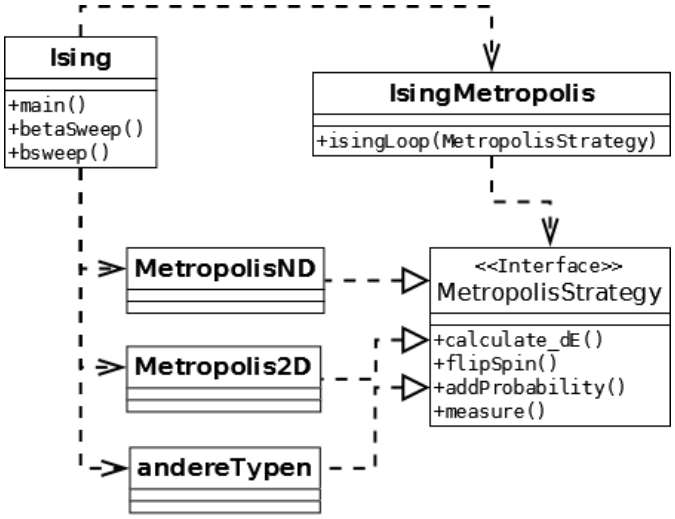


Abbildung 1: Darstellung des Strategie-Patterns

Der Vorteil dieses Designpatterns liegt darin, dass die Strategie, in unserem Fall die Struktur der Probe, austauschbar ist, ohne die Implementierung des Algorithmus' zu verändern; es muss lediglich eine neue Klasse das Strategie-Interface implementieren.

3.2 Simulationsstart

In *Ising.cpp* werden die Parameter eingelesen. Sie können entweder über die Konsole oder über eine Konfigurationsdatei angegeben werden. Je nach Angabe der Parameter kann entschieden werden was für eine Simulation (B-Feld-Sweep, Temperatur-Sweep) mit welcher Strategie gestartet werden soll. Zudem werden die Ausgaben der Simulation gesteuert.

Beim Temperatur-Sweep wird $\beta = \frac{1}{k_B T}$ (wobei $k_B = 1$) im angegebenen Intervall in äquidistanten Schritten variiert. Da jede Temperatursimulation von den anderen unabhängig ist konnte diese Messreihe leicht mit *OpenMP* parallelisiert werden. Dazu wird für jeden Thread eine Kopie der Strategie erstellt (*clone()*) und die for-Schleife mit *#pragma omp for* parallelisiert. Es werden für jede Temperatur ein Bild mit der letzten Spin-Konfiguration und eines mit den Mittelwerten ausgegeben. Die Messewerte (β, mittlere Magnetisierung pro Spin und mittlere Fliprate) werden in der Konsole ausgegeben.

Mit B-Feld-Sweep lässt sich eine Hysterese-Kurve aufzeichnen. Dazu wird mit der selben Strategie B-Felder (in äquidistanten Schritten) durchlaufen:

$$0 \to B_{max} \to B_{min} \to B_{max}$$

Da nun aber jeder Schritt (im B-Feld) der Simulation auf dem vorherigen aufbaut, ließ sich dies nicht mit *OpenMP* parallelisieren. Es werden ebenfalls Bilder für jeden Schritt ausgegeben und die Messwerte (B, mittlere Magnetisierung pro Spin und mittlere Fliprate) in die Konsole geschrieben.

3.3 Implementation des Algorithmus'

Weil das Strategie-Pattern verwendet wird, definiert *IsingMetropolis.cpp* nur die Struktur, die bereits in Abschnitt 2 beschrieben ist. Die Berechnung der Energiedifferenz, das Flippen der Spins, die Wahrscheinlichkeitszählung und die Messung sind ausgelagert. Nur das Bestimmen des zufälligen Spins und die Flipwahrscheinlichkeit (nach (3)) sind nicht ausgelagert. Des weiteren wird die Anzahl der Flips mit gezählt, um die Aktivität des Systems erfassen zu können.

3.4 Die Metropolis-Strategie

Die Schnittstelle der Strategie ist in *MetropolisStrategy.h* beschrieben. Für den Metropolis-Algorithmus wichtigen Funktionen sind:

spinNumber() zur Bestimmung der Spinanzahl (Maximum für Zufallsspin)

calculate_dE(int i) Bestimmt die Energiedifferenz, wenn Spin i flippte

flipSpin(int i) Flippt den Spin i

addProbability(int i) Zeichnet den Wert des Spins i auf (nach (4))

measure() Vermisst das System über den Erwartungswert (nach (5))

Die restlichen Methoden dienen zur Simulationssteuerung (*reset()* , *resetMeasure()* , *setBField(double)* , *clone()*) bzw. zur Bildausgabe.

Wir verwenden nur eine Strategie, da sie die anderen darstellen kann und andere (noch) nicht implementiert sind: *MetropolisND*. Es ist aber denkbar, dass z.B. eine Strategie mit Störstellen, eine anderer Gitteraufteilung (z.B. eine hexagonale Struktur) oder nicht nur nächste-Nachbarn-Näherung hinzugefügt werden.

MetropolisND Sowohl der Zugriff von außen (um die Schnittstelle allgemein zu halten) als auch die Speicherung der Spins (siehe Abschnitt 3.5) erfolgt in linearer Weise. Da aber die nächste-Nachbar-Wechselwirkung in alle Dimensionen verläuft, ist eine Transformation des linearen Index' in eine ND-Koordinate notwendig. Zur Transformation wird einen Abwandlung des "Turning-Weels"-Algorithmus' (wird verwendet um ein Kartesisches Produkt zu bilden, unser Algorithmus nummeriert quasi diese Elemente) verwendet. Der Index berechnet sich durch

$$i = \sum_{j=1}^{D} c(j) \prod_{d=1}^{j-1} s(d) \tag{6}$$

wobei D die Anzahl der Dimensionen angibt, $c(j)$ der Wert der Koordinate j und $s(d)$ die Größe in der Dimension d ist. Zum Berechnen der Koordinate wird die Ganzzahldivision und der Modulo verwendet:

$$c(j) = \frac{i}{\prod_{d=1}^{j-1} s(d)} \mod s(j) \tag{7}$$

Die Berechnung der Energiedifferenz nach Gleichung (2) ist aufwendig, da sowohl die Gesamtenergie des Systems S als auch S' berechnet werden muss. Betrachtet man nun den Hamiltonoperator (1), läge die Laufzeit der Berechnung in $\mathcal{O}(N^2)$, wobei N die Anzahl der Spins ist; betrachtet man nur noch die nächste-Nachbar-Wechselwirkung, ist diese nur noch in $\mathcal{O}(N \cdot z)$, wobei z die Anzahl der nächsten Nachbarn ist. Da sich zum einen nur eine Spin verändert, nur die nächsten Nachbarn betrachtet werden, $J_{ij} = J$ konstant für das ganze System ist und nicht die Einzelenergien sondern die Energiedifferenz benötigt werden, vereinfacht die Berechnung der Energiedifferenz:

$$\Delta E = J \Delta S_i \sum_{j=n.n.} S_j + B \Delta S_i \tag{8}$$

$$\Delta S_i = S_i - S_i' = \pm 2 \tag{9}$$

Die Laufzeit ist nun in $\mathcal{O}(z)$ und hängt somit nicht mehr von der Größe des Systems ab (sofern man nicht die Anzahl der Dimensionen erhöht). Daher ist ein Berechnungsschritt für ein großes System genauso teuer wie das eines Kleinen.

3.5 Spin-Array

Hierbei handelt es sich um eine effiziente Datenstruktur, die es erlaubt die Spin-Konfiguration mit wenig Speicherbedarf zu verwalten.

Da jeder Spin im Ising-Modell nur 2 Werte annehmen kann, ist es möglich einen Spin mit einem einem Bit zu codieren. Daher besteht die Datenstruktur aus einem Bytearray, dessen Bits die Spins halten.

Es werden 3 einfache Methoden definiert: Zum Auslesen, Setzen und Flippen eines Spins. Alle drei Methoden sind ähnlich aufgebaut: Zuerst wird das Byte und dann das Bit dieses Bytes bestimmt, das den gewünschten Spin enthält, dann wir eine Bitoperation (and, or, xor) auf dieses Bit angewendet und entweder das neue Byte im Array gespeichert oder den Wert des Spins zurückgegeben.

4 Simulations-Ergebnisse

4.1 Der kritische Punkt

Ferromagneten zeigen bei einer bestimmten Temperatur einen Phasenübergang. Wenn die Temperatur einen kritischen Wert unterschreitet, kann eine spontane Magnetisierung einsetzen. Das Ising-Modell hat diesen Phasenübergang bei zwei Dimensionen und mehr. In einer Dimension gibt es zu wenige nächste Nachbarn.

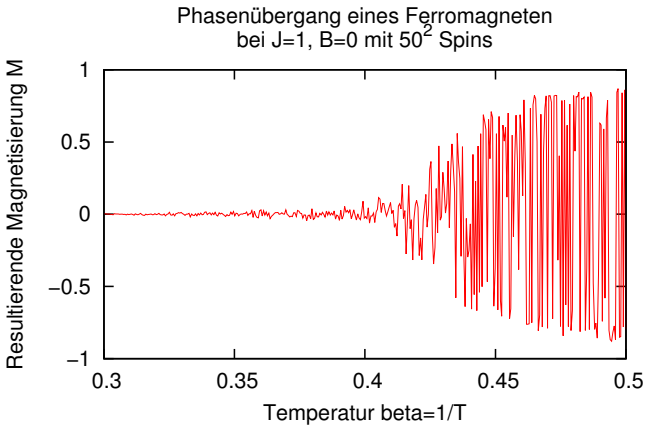


Abbildung 2: Phasenübergang eines Ferromagneten. Die Simulation ging mit 400 Schritten von $\beta = 0.3$ bis $\beta = 0.5$

Abbildung 2 zeigt, wie ab einer Temperatur von $\beta = 1/T \approx 0.4$ die spontane Magnetisierung einsetzt. Die Simulation wurde mit 50^2 Spins, ohne Magnetfeld und für jedes der 400 verschiedenen β mit $6 \cdot 10^6$ Monte-Carlo-Schritten durchgeführt. Man kann gut sehen, dass die Magnetisierung mit zunehmendem β immer stärker ausschlägt und bei fast jeder Messung annähernd die Sättigung erreicht wird. Dabei ist es vollkommen zufällig, in welche Richtung der Magnet ausgerichtet ist.

4.2 Hysterese beim Ferromagneten

Ein Ferromagnet widersteht einem wechselnden äußeren Magnetfeld, bevor er selbst seine Ausrichtung ändert. Diese Hysterese lässt sich auch mit dem Ising-Modell gut beschreiben.

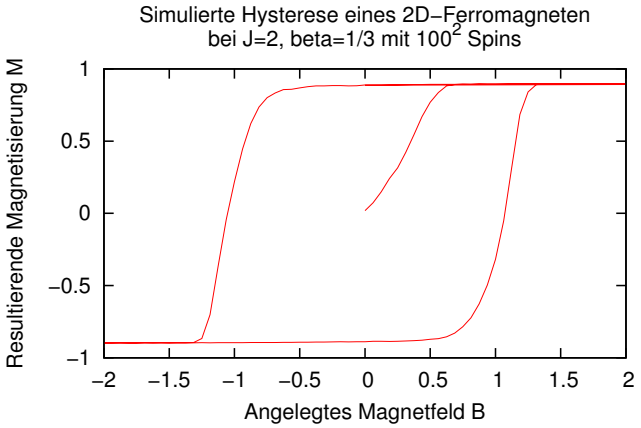


Abbildung 3: Hysterese eines Ferromagneten. Die Simulation ging mit 400 Schritten: $B = 0 \to 5 \to -5 \to 5$

Für die Simulation haben wir wieder 100^2 Spins mit einer ferromagnetischen Wechselwirkung von $J = 2$. Die Temperatur ist mit $\beta = \frac{1}{3}$ klein genug für die ferromagnetische Phase. Das externe Magnetfeld läuft von $B = 0$ bis zur Sättigung $B = 5$, anschließend bis $B = -5$ und noch einmal ins Positive. Dabei wird B in 400 Einzel-

schritten linear geändert und mit jeweils 10^5 Monte-Carlo-Schritten stabilisiert. Das Ergebnis (Abbildung 3) zeigt einen relativ glatten Verlauf und die Rechenzeit hielt sich mit 21 Sekunden in Grenzen, so dass wir bislang keine weitere Geschwindigkeitsoptimierung vorgenommen haben.

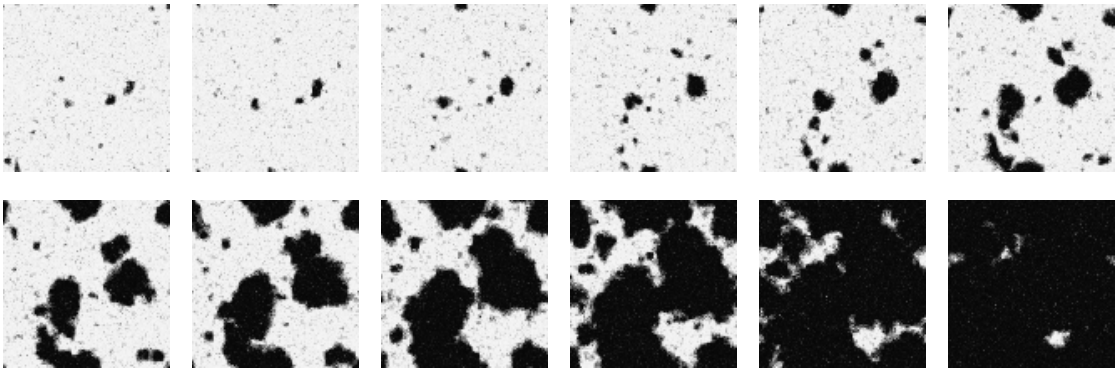


Abbildung 4: Ausrichtungswahrscheinlichkeiten der Spins beim Richtungswechsel der Hysteresekurve.

Der Übergang von positiver zu negativer Magnetisierung der Hysteresekurve ist nochmal in Bildern in Abbildung 4 zu sehen. Hier sind die Mittelwerte der einzelnen Spins (ein Spin pro Pixel) wie in Abbildung 5 (a) zu sehen. Man kann gut die zufälligen Strukturen erkennen, die sich beim Übergang ergeben. Bestehende Flächen von Spins, die schon negativ (schwarz) ausgerichtet sind, werden immer größer und breiten sich immer weiter aus.

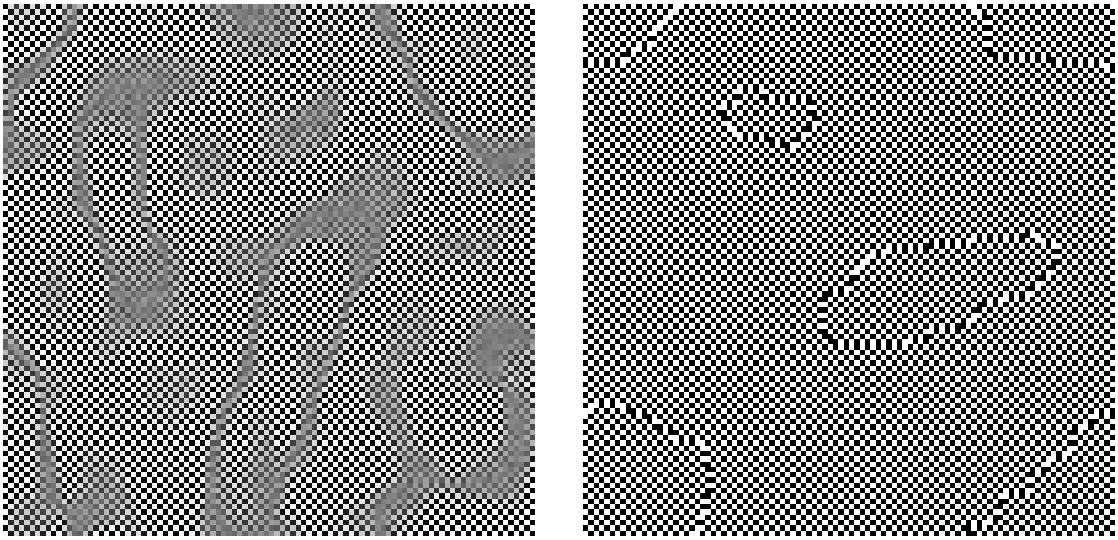
4.3 Domänenwände beim Antiferromagneten

Bei geringen Temperaturen gibt es in einem Antiferromagneten nur noch sehr wenige Störstellen. Da sich das *Schachbrettmuster* beim Abkühlen von mehreren Stellen gleichzeitig ausbreitet, bleiben am Ende einige Stellen übrig, an denen es unterbrochen ist. Diese Domänenwände haben eine Breite von genau zwei Spins, die nebeneinander die gleiche Ausrichtung haben. Es wäre energetisch viel aufwändiger, einen ganzen Bereich mit *falscher* Ausrichtung umzudrehen, als diese dünnen Wände stehen zu lassen.

Obwohl die Simulation bei uns mit konstanter Temperatur abläuft, wird praktisch eine Abkühlung simuliert. Die Startkonfiguration ist eine Zufallsverteilung, die einer hohen Temperatur entspricht.

Abbildung 5 zeigt eine Beispielsimulation, bei der Domänenwände sichtbar werden. Simuliert wurde in zwei Dimensionen mit 100^2 Spins und ohne externes Magnetfeld. Die Simulation ist nur eingeschränkt auf die Realität übertragbar, da die Grundannahme des Ising-Modells, dass Spins nur in einer Achse ausgerichtet sein können, ohne externes Magnetfeld natürlich nicht erfüllt ist.

Bei den Abbildungen weichen die Mittelwerte doch an einigen Stellen deutlich von der Momentaufnahme ab. Um hier wirklich ein statisches Bild zu geben, scheint die



(a) Mittelwerte der Spins (b) Letzte Spin-Konfiguration der Messung

Abbildung 5: Domänenwände eines Antiferromagneten bei $\beta \approx 345, J = -1, B = 0$

Temperatur noch nicht niedrig genug zu sein.