Appell an die wissenschaftliche Ehrlichkeit:
Schreiben Sie Ihre Programme und Diskussion eigenständig!

Computational Physics
Sommersemester 2024
Blatt 10, Abgabe: Montag, 08. Juli, 8:00

Institut für Theoretische Physik
Prof. Dr. A. Bäcker

Ising-Modell

Aufgabe 10.1: **(16 Punkte)**

Visualisieren Sie das zweidimensionale Ising-Modell, indem Sie in einem linken Plot-Bereich den Spinzustand eines 50×50 Gitters (mit periodischen Randbedingungen) darstellen und in einem rechten Plot-Bereich die mittlere Magnetisierung m pro Spin über der dimensionslosen Temperatur τ anzeigen.

- Zeichnen Sie in den rechten Plot-Bereich die analytische Vorhersage m_∞ als Funktion von τ ein.
- Mittels Mausklick im rechten Plot-Bereich sollen m und τ neu gewählt werden können. Schreiben Sie eine Funktion, die einen Spinzustand mit zufälligen unabhängigen Spins erzeugt, so dass die mittlere Magnetisierung pro Spin ungefähr dem gewählten m entspricht. Stellen Sie den tatsächlichen Wert von m für den erzeugten Spinzustand als Punkt im rechten Plot-Bereich dar und visualisieren Sie den Spinzustand im linken Plot-Bereich.
- Nach Mausklick im linken Plot-Bereich sollen beginnend vom aktuellen Spinzustand 10 Monte-Carlo-Zeitschritte durchgeführt und der jeweilige Spinzustand (links) und das aktuelle m (rechts) dynamisch dargestellt werden. Schreiben Sie hierzu eine Funktion, die einen Monte-Carlo-Schritt ausführt.

  Bei erneutem Mausklick soll diese dynamische Darstellung für weitere 10 Schritte fortgesetzt werden.
- Das Programm soll vor dem ersten Mausklick einen Spinzustand mit $m \approx 0{,}6$ zeigen und für $\tau = 1{,}5$ eingestellt sein, so dass bereits mit Mausklick im linken Plot-Bereich die Monte-Carlo-Schritte gestartet werden können.
- Beschreiben Sie im abschließenden Kommentar qualitativ das Verhalten für die Werte $\tau = 1{,}5$ und $\tau = 3{,}5$ und verschiedene Startwerte von m.

Vorgaben und Hinweise:

- Periodische Randbedingungen lassen sich z.B. wie folgt implementieren: für eine $n \times n$-Matrix ist der rechte Nachbar des Matrixelements `matrix[i, j]` durch den Eintrag `matrix[i, (j+1) % n]` gegeben.
- Während eines Monte-Carlo-Zeitschritts werden n^2 Gitterpunkte (zufällig gewählt) nach dem Metropolis–Algorithmus auf mögliches Umklappen des Spins getestet.
- Um zu entscheiden, in welchen der vorhandenen Plotbereiche (z.B `ax_1`, `ax_2`) geklickt wurde, nutzt man folgende Abfrage:

```
if event.inaxes == ax_2:
    # Berechnung starten ...
```

- Visualisieren Sie die Spinmatrix mit `ax.imshow(...)`.

- Verwenden Sie zum Speichern des Zustands des Systems, gegeben durch die Matrix der Spins und der dimensionslosen Temperatur, eine Instanz einer selbstdefinierten Klasse als "Datencontainer":

```
class IsingZustand:
    """Klasse zur Speicherung des Zustands und der Temperatur."""
    def __init__(self, spins, tau):
        """Initialisierung: ‘spins‘ + Temperatur ‘tau‘."""
        self.spins = spins
        self.tau = tau
```

  Diese wird im Hauptprogramm wie folgt genutzt:

```
def maus_klick(..., zustand, ...):
    # Nutzung von zustand.spins und zustand.tau
    ...

def main():
    tau = 1.0
    spins_matrix = ...
    ising_zustand = IsingZustand(spins_matrix, tau)

    spins_img = ax.imshow(ising_zustand.spins, ...)
    ...
    klick_function = functools.partial(maus_klick,
                                       ax=ax,
                                       zustand=ising_zustand,
                                       spins_img=spins_img, ...)
    fig.canvas.mpl_connect('button_press_event', klick_function)
```

  In der Event-Funktion `maus_klick` kann dann auf `zustand.tau` und `zustand.spins` zugegriffen werden, z.B. um neue Werte zu setzen.

- Zur dynamischen Darstellung der Spinmatrix verwenden Sie in der Event-Funktion den Befehl `spins_img.set_data(zustand.spins)` und das Konstrukt `event.canvas.flush_events()` gefolgt von `event.canvas.draw()`.