1 HW 3: Applying HMC to the Long-Range Ising Model

Your total number of points for this homework is (16/20 P). If you have any questions feel free to write me an email (s6nnschl@uni-bonn.de).

Exercise 3.1: Observable Derivations

Your result $m[\phi] = \tanh(\beta h + \phi)$ is correct. In the energy case you forgot to absorb the minus sign into the parenthesis (cf. second last line of your calculation). So here the correct result would be

$$\epsilon[\phi] = -\frac{\phi^2}{2\beta^2 J} - h\,\tanh(\beta h + \phi) + \frac{1}{2\beta N}. \tag{1}$$

In total (0.5/1 P)

Exercise 3.2: Equations of Motion

Correct (1/1 P)

Exercise 3.3: Leapfrog Algorithm and Convergence

The leapfrog algorithm is correct and your plot shows the expected convergence behavior. In total (3/3 P)

Exercise 3.4: HMC Algorithm

Exercise 3.4.1: Change in Action

Correct (2/2 P)

Exercise 3.4.2: p Sampling

Correct that you sample the momenta from a normal distribution. (1/1 P)

Exercise 3.4.3: p Update

Correct (1/1 P)

Exercise 3.4.4: ϕ Update

(0/2 P): Here your HMC algorithm fails because you always set q_0 to 1 inside the for loop. However, you should initialize only once outside the loop and then set q_0 = q_1 in the accept step. With the following corrections and an appropriate choice of input parameters, I was able to get pretty close to the exact solution with your code.

```python
def HMC(N_s,N_md,J,h,N):

    q_mc = np.ones(N_s)
    p_mc = np.ones(N_s)

    acc = 0

    q_0 = 1.0

    for i in range(N_s):

        p_0 = np.random.normal()

        p_l,q_l = leapfrog(N_md,p_0,q_0,J,h,N)

        P_0 = P_acc(p_0,q_0,J,h,N)
        P_l = P_acc(p_l,q_l,J,h,N)

        r = np.random.uniform(0,1)

        if P_l>P_0:
            q_mc[i] = q_l
            p_mc[i] = p_l
            q_0 = q_l
            acc += 1

        elif P_l/P_0>r:
            q_mc[i] = q_l
            p_mc[i] = p_l
            q_0 = q_l
            acc += 1

        else:
            q_mc[i] = q_0
            p_mc[i] = p_0


    return q_mc,p_mc,acc/N_s
```

Exercise 3.4.5: Accept-Reject Method

Here, you have to sample r from a uniform distribution, i.e. `np.random.uniform(0,1)`. (1.5/2 P)

Exercise 3.4.6: Saving Accepted Configurations

Works fine. (2/2 P)

Exercise 3.5: Final Measurements

Exercise 3.5.1: Mean Magnetization per Lattice Site

In principle the computation of the magnetization per site is correct. I used your code with the above corrections and was able to create a correct plot. (2/2 P)

Exercise 3.5.2: Energy per Lattice Site

In principle the computation of the energy per site is correct. I used your code with the above corrections and was able to create a correct plot. (2/2 P)

Exercise 3.5.3: N_{md} Tuning

$N_{md} = 100$ is much too large. You can see that in the acceptance rate, which is very close to 100% leading to large autocorrelation. (0/1 P)