Heisenberg model

The Heisenberg model generalises the Ising model, where, instead of binary values, the spins are 3-component vector valued and take values on the surface of the unit sphere, i.e. $\vec{S}_i \in \mathbb{S}^2 \subset \mathbb{R}^3$. *With* $J > 0$, *the Hamiltonian penalises misalignment of nearest-neighbour spins:*

$$\mathcal{H} = -J \sum_{\langle i,j \rangle} \vec{S}_i \cdot \vec{S}_j. \tag{1}$$

Since we have no magnetic field, we set $J = 1$ *for the remainder of this sheet.*

While for the XY model ($\vec{S}_i \in \mathbb{S}^1 \subset \mathbb{R}^2$*) it is advantageous to use polar coordinates, it not clear if in the 3-component spin case the added complexity and the additional floating point operations justifies the memory savings.*

Task 1: *Implement a Metropolis simulation of the 3D Heisenberg model. Except for a few changes in the update probability computation and the representation of your cluster, you should not need to drastically modify your code for the Ising model. Remember to normalize your vectors if you use a Cartesian representation.*

Task 2: *The cluster algorithms (Swendsen-Wang or Wolff) can be extended to systems with any number of spin components by considering reflections around a random plane at each MC step.*

1. *Select a random reflection plane by drawing a random normal unit vector* $\hat{n}$. *Select a random lattice site* i *and reflect the spin* $\vec{S}_i \leftarrow R_{\hat{n}}(\vec{S}_i) = \vec{S}_i - 2\hat{n}(\vec{S}_i \cdot \hat{n})$.
2. *Grow a cluster with bond probability*

$$p_c(i,j) = 1 - \exp\left\{\min\left[0, 2\beta\left(\vec{S}_i \cdot \hat{n}\right)\left(\vec{S}_j \cdot \hat{n}\right)\right]\right\}, \tag{2}$$

   where $\beta = 1/k_\mathrm{B}T$. *Note that* $\vec{S}_i$ *is the reflected spin in step 1.*
3. *In the case of acceptance, flip the neighbouring spin as* $\vec{S}_j \leftarrow R_{\hat{n}}(\vec{S}_j)$ *and add it to the cluster.*

Plot your results for the order parameter of this system, which is the spontaneous magnetisation density:

$$\langle |\vec{m}| \rangle = \langle m \rangle = \left\langle \frac{1}{L^3} \left| \sum_{i=1}^{L^3} \vec{S}_i \right| \right\rangle. \tag{3}$$

Also plot the energy density $\langle \mathcal{H} \rangle / L^3$ *and the response functions: magnetic susceptibility* χ *and specific heat* c_V *as a function of* T *for different* L. *Compare the size of magnetisation fluctuations to the Ising model by inspecting* χ *and physically interpret your observation.*

Task 3: *Estimate the critical temperature* T_c *using the Binder cumulant for order parameter* m

$$U_4(T, L) = 1 - \frac{\langle m^4(T) \rangle}{3\langle m^2(T) \rangle^2}. \tag{4}$$

Hint: Recall that $U_4(T_c, L) = U_4(T_c)$. *You can use small system sizes.*

Task 4 (optional): *Compute the autocorrelation time* τ *for different* L, *either for* E *or* m, *at* T_c *and find the dynamical critical exponent* z_c, *which describes the algebraic finite-size scaling* $\tau \sim L^{z_c}$, *by finding the best fit to your data.*

Solution. By the Mermin-Wagner-Hohenberg theorem we know that there is no symmetry breaking phase transition at dimensions 2 and higher for spins with continuous symmetry. Since the $n = 2$ case (the XY model) at 2-dimensions is already marginal (*i.e.* Berezinskii-Kosterlitz-Thouless transition), this suggests that the interesting situation for $n = 3$ (the Heisenberg model) is in 3-dimensions.

Tasks 1, 2: Implementing the Monte Carlo algorithms.

Metropolis-Hastings algorithm: To implement the Metropolis algorithm for the Heisenberg model, one first needs to decide the coordinate system in which the vector spin $\vec{S} \in \mathbb{S}^2$ is encoded. Here we will use Cartesian coordinates.[1] In order to sample points on the surface of the unit 2-sphere, we first sample three points s_1, s_2, s_3 from the standard normal distribution

$$p(s) = \frac{1}{\sqrt{2\pi}} e^{-s^2/2}, \tag{S.1}$$

and then normalise the vector $\vec{s}$ to get a valid sample spin $\vec{S}$ at infinite temperature.[2]

We initialise our simulation effectively at infinite temperature by generating a random spin configuration, by sampling spins at each site as described above. The local Metropolis updates proceed by randomly selecting a lattice site i and proposing a new orientation $\vec{S}_i' \to \vec{S}_i$ for the spin. The acceptance probability is then computed using the energy cost of this update

$$\Delta E_i = \sum_{j \in \mathrm{nn(i)}} \left[(\vec{S}_i' - \vec{S}_i) \cdot \vec{S}_j \right]$$

and is given at inverse temperature β by

$$p_i^{\mathrm{a}} = \min\left(1, e^{-\beta \Delta E_i}\right). \tag{S.2}$$

The spin proposals can be made as a uniform sampling over $\mathbb{S}^2$. On the other hand, a more efficient approach is to take advantage of the continuous symmetry and propose updates that are only small deformations of the current spin, so that the acceptance probability would be higher.

The specifications for our simulation with local updates are as follows: we thermalise the random initial configuration by making $100 \times L^3$ updates (*i.e.* 100 sweeps). We then take 10^4 samples by measuring the thermodynamic quantities after one sweep.

Wolff cluster algorithm: The trick for implementing the Wolff algorithm for spins with continuous symmetry is to generalise the spin flip operation as a reflection with respect to a random mirror plane. More concretely, we first select a random site and we draw a random normal vector $\hat{n}$ over $\mathbb{S}^2$ as described above. The spin at site i is then flipped as

$$R_{\hat{n}} : \vec{S}_i \mapsto \vec{S}_i - 2\hat{n}(\vec{S}_i \cdot \hat{n}). \tag{S.3}$$

Next, we attempt to grow a cluster from seed i by visiting all links connecting to i to its nearest neighbours j. The activation (or acceptance) probability for the bond $\langle i, j \rangle$ is defined analogously

[1] For the polar coordinates, one can sample uniformly over $\mathbb{S}^2$ by recalling the form of the differential solid angle $\mathrm{d}\Omega = -\mathrm{d}\phi\, \mathrm{d}\cos\theta$.

[2] Note that sampling the components of the spin uniformly within $[-1, 1]$ would not work as this would sample spins on the surface of the cube and not the sphere.

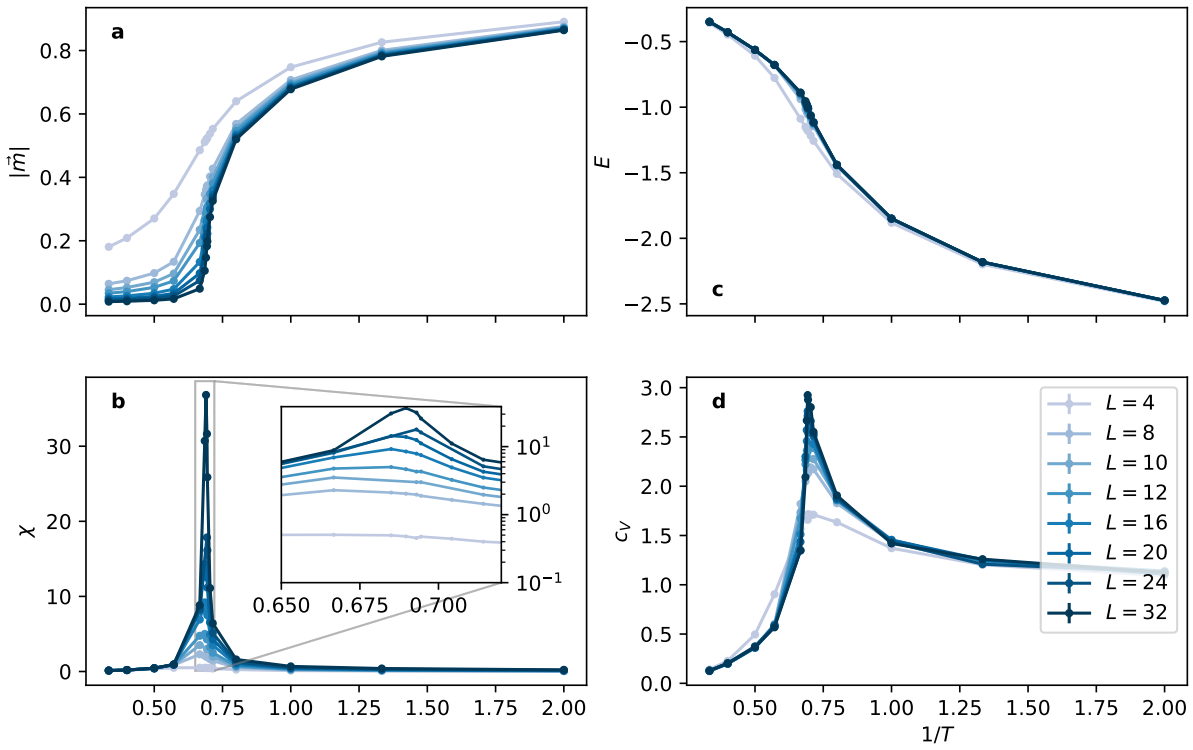

Figure 1: **a** The order parameter for the Heisenberg model is the modulus of the magnetisation density. It has a kink around $\beta = 1/T \approx 0.7$ that gets sharper with increasing system size, suggesting a phase transition from a paramagnetic to a magnetised state. However, one needs to be cautious regarding the existence of the long-range ordered phase in the thermodynamic limit since there is a slight decay of the magnitude of the order parameter. **b** Magnetic susceptibility peaks at $\beta_c \approx 0.7$. **c** The behaviour of the energy density across the phase transition is analogous to the Ising transition. **d** The specific heat also seems to develop a non-analyticity near $\beta_c \approx 0.7$.

as in the Ising case through the energy cost of flipping spin at site j:

$$p^c_{\langle i,j\rangle} = 1 - \exp\left\{\beta \min\left[0, \vec{S}_i \cdot \vec{S}_j - \vec{S}_i \cdot R_{\hat{n}}\left(\vec{S}_j\right)\right]\right\} \tag{S.4}$$

$$= 1 - \exp\left\{2\beta \min\left[0, \left(\hat{n} \cdot \vec{S}_i\right)\left(\hat{n} \cdot \vec{S}_j\right)\right]\right\}. \tag{S.5}$$

In case of acceptance, the site j is joined to the cluster. Then we repeat until the cluster cannot grow any longer.

As for the Metropolis algorithm, we start our cluster simulation by initialising at a random spin configuration and thermalising by making 100 sweeps. We again take 10^4 measurements. The resulting thermodynamic estimators for linear system sizes $L = 4, 8, 10, 12, 16, 20, 32$ are shown in Fig. 1. We also applied jackknife analysis on the measurement samples.

Task 3: Approximating the phase transition point.

We show in Fig. 2 the estimators for the Binder cumulant at different system sizes obtained by the Wolff algorithm. We approximate the phase transition point by locating the crossing at $1/T_c = \beta_c \approx 0.693$. Note that, as for the other thermodynamic quantities shown above, we use jackknife resampling on cumulants of m to get a measure for bias and variance in the statistics.

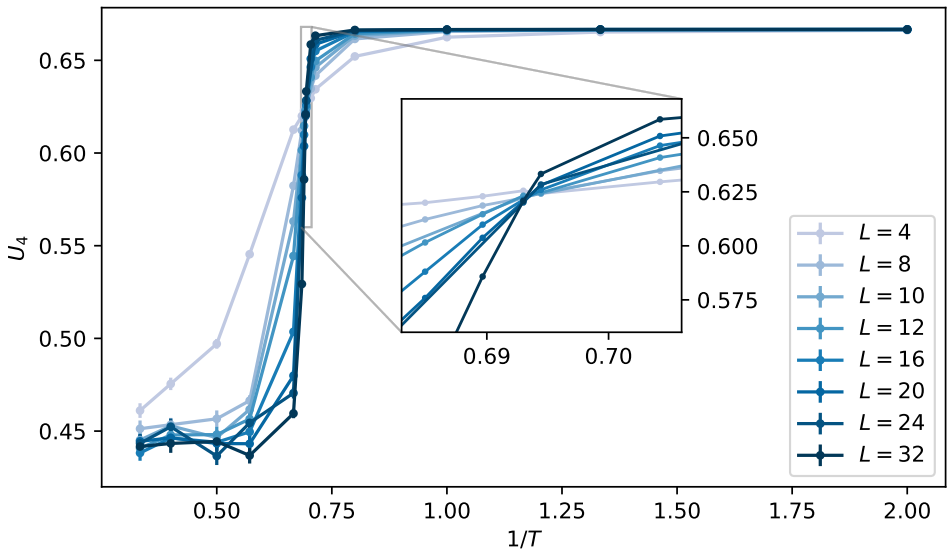

Figure 2: There is a crossing of the Binder cumulants for different linear system sizes L at $\beta_c \approx 0.693$. The error bars (at most points are smaller than the markers) indicate the variance obtained by jackknife analysis.

Once we get the critical point, we can estimate the critical exponents by means of the finite-size scaling ansatz and find data collapse, as shown in Fig. 3 for γ and ν. Note that it suffices to determine two of the exponents and the remaining 4 can be calculated using the scaling laws.

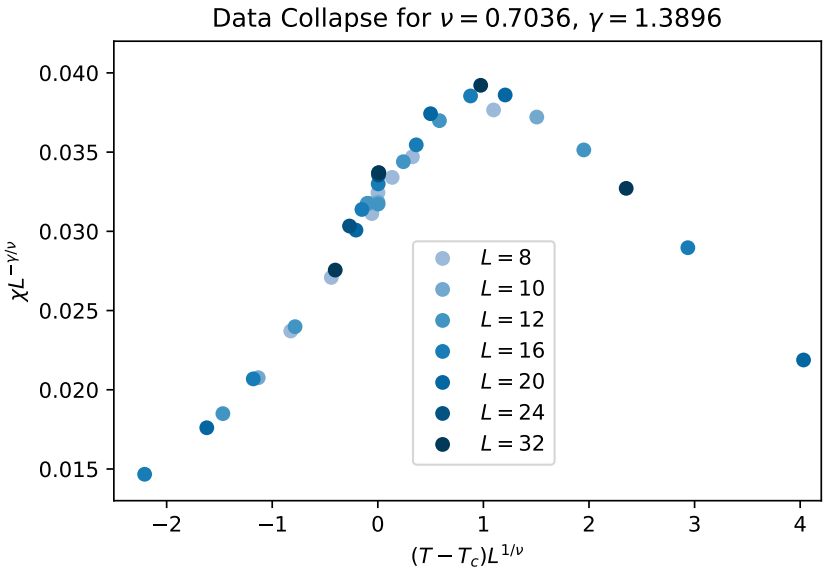

Figure 3: Data collapse of magnetic susceptibility with the literature values of critical exponents $\nu = 0.7036(23)$ and $\gamma = 1.3896(70)$.

Task 4: Autocorrelation analysis.

We tabulate in Tabs. 1,2 the scaling of the runtime, the autocorrelations and the MC speed with the linear system size L for the Metropolis and the Wolff algorithm, respectively. We find that although the runtime per measurement is similar in both algorithms (sweeping the whole lattice in Metropolis vs forming a large cluster in Wolff), it can be seen by comparing the autocorrelations that the Wolff algorithm largely reduces the critical slowing down.

Table 1: Scaling of autocorrelation and runtimes of the Metropolis-Hastings algorithm for the 3-dimensional Heisenberg model at the critical point $\beta_c \approx 0.693$.

L	runtime [s/sweep]	autocorrelation time	MC speed
4	8.09325e-06	14.8111	8342.35
8	4.90628e-05	38.6865	526.852
16	0.000398978	130.058	19.2714
32	0.00318075	216.156	1.45447

Table 2: Scaling of autocorrelation and runtimes of the Wolff algorithm for the 3-dimensional Heisenberg model at the critical point $\beta_c \approx 0.693$.

L	runtime [s/sweep]	autocorrelation time	MC speed
4	1.13991e-05	4.04575	21683.5
8	0.000103191	4.29137	2258.2
16	0.000622483	5.1113	314.298
32	0.00583858	5.67847	30.1621

For the Metropolis-Hastings algorithm one can use a scaling *ansatz* for the autocorrelation time τ of the energy

$$\tau_E = \int_0^\infty \Phi_E(t)\mathrm{d}t, \tag{S.6}$$

where $\Phi_E(t) = \left[\langle E(t_0)E(t)\rangle - \langle E(t_0)\rangle^2\right] / \left[\langle E^2(t_0)\rangle - \langle E(t_0)\rangle^2\right]$ is the autocorrelation function for the energy. Using this, we obtain an estimate for the dynamical critical exponent of τ via linear regression on $\log L$ vs $\log \tau$ data. As shown in Figs. 4 and 5, we get $z_c = 1.335$ for Metropolis and $z_c = 0.172$ for Wolff.

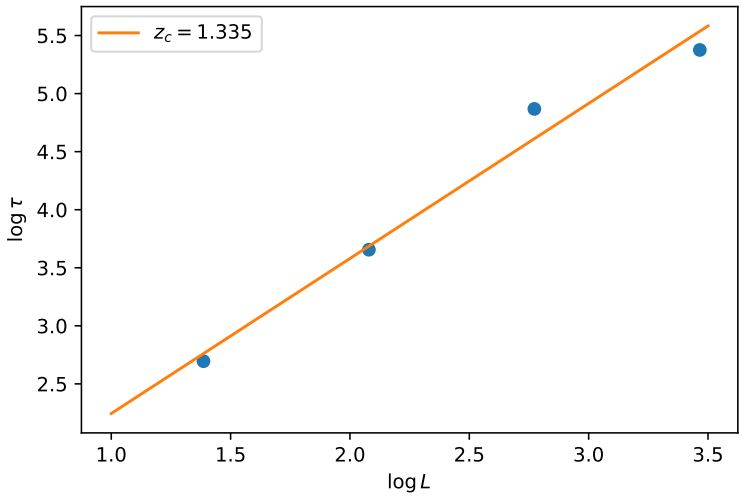

Figure 4: Best fit for the dynamical critical exponent of the Metropolis-Hastings algorithm for the 3-d Heisenberg model is $z_c = 1.335$.

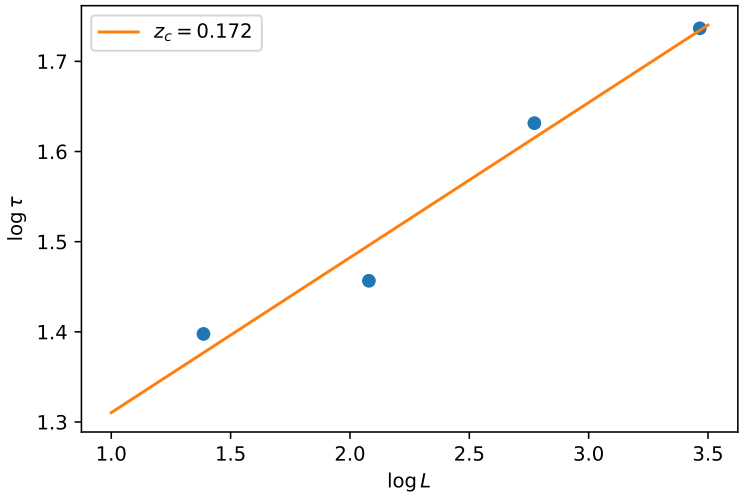

Figure 5: Best fit for the dynamical critical exponent of the Wolff algorithm for the 3-d Heisenberg model is $z_c = 0.172$.