Exercises

12.13 **Hysteresis and avalanches: scaling.**[1] (Complexity) ③

For this exercise, either download Matt Kuntz's hysteresis simulation code from the book web site [129], or make use of the software you developed in Exercise 8.13 or 8.14.

Run the simulation in two dimensions on a 1000×1000 lattice with disorder $R = 0.9$, or a three-dimensional simulation on a 100^3 lattice at $R = 2.16$.[2] The simulation is a simplified model of magnetic hysteresis, described in [128]; see also [127]. The spins s_i begin all pointing down, and flip upward as the external field H grows from minus infinity, depending on the spins of their neighbors and a local random field h_i. The flipped spins are colored as they flip, with spins in the same *avalanche* sharing the same color. An avalanche is a collection of spins which flip together, all triggered from the same original spin. The *disorder* is the ratio R of the root-mean-square width $\sqrt{\langle h_i^2 \rangle}$ to the ferromagnetic coupling J between spins:

$$R = \sqrt{\langle h^2 \rangle}/J. \tag{1}$$

Examine the $M(H)$ curve for our model and the $\mathrm{d}M/\mathrm{d}H$ curve. The individual avalanches should be visible on the first graph as jumps, and on the second graph as spikes. This kind of time series (a set of spikes or pulses with a broad range of sizes) we hear as *crackling noise*. You can go to our site [68] to hear the noise resulting from our model, as well as crackling noise we have assembled from crumpling paper, from fires and Rice Krispies™, and from the Earth (earthquakes in 1995, sped up to audio frequencies).

Examine the avalanche size distribution. The (unlabeled) vertical axis on the log–log plot gives the number of avalanches $D(S, R)$; the horizontal axis gives the size S (with $S = 1$ on the left-hand side). Equivalently, $D(S, R)$ is the probability distribution that a given avalanche during the simulation will have size S. The graph is created as a histogram, and the curve changes color after the first bin with zero entries (after which the data becomes much less useful, and should be ignored).

If available, examine the spin–spin correlation function $C(x, R)$. It shows a log–log plot of the probability (vertical axis) that an avalanche initiated at a point $\mathbf{x}_0$ will extend to include a spin $\mathbf{x}_1$ a distance $x = \sqrt{(\mathbf{x}_1 - \mathbf{x}_0)^2}$ away.

Two dimensions is fun to watch, but the scaling behavior is not yet understood. In three dimensions we have good evidence for scaling and criticality at a phase transition in the dynamical evolution. There is a phase transition in the dynamics at $R_c \sim 2.16$ on the three-dimensional cubic lattice. Well below R_c one large avalanche flips most of the spins. Well above R_c all avalanches are fairly small; at very high disorder each spin flips individually. The critical disorder is the point, as $L \to \infty$, where one first finds *spanning avalanches*, which extend from one side of the simulation to the other.

Simulate a 3D system at $R = R_c = 2.16$ with $L = 100$ (one million spins, or larger, if you have a fast machine). It will be fastest if you use the *sorted list* algorithm (Exercise 8.14). The display will show an $L \times L$ cross-section of the 3D avalanches. Notice that there are many tiny avalanches, and a few large ones. Below R_c you will find one large colored region forming the background for the others; this is the spanning, or *infinite* avalanche. Look at the $M(H)$ curve (the bottom half of the hysteresis loop). It has many small vertical jumps (avalanches), and one large one (corresponding to the spanning avalanche).

(a) *What fraction of the system is flipped by the one largest avalanche, in your simulation? Compare this with the hysteresis curve at $R = 2.4 > R_c$.*

[1] From *Statistical Mechanics: Entropy, Order Parameters, and Complexity* by James P. Sethna, copyright Oxford University Press, 2007, page 296. A pdf of the text is available at pages.physics.cornell.edu/sethna/StatMech/ (select the picture of the text). Hyperlinks from this exercise into the text will work if the latter PDF is downloaded into the same directory/folder as this PDF.

[2] If you are using the brute-force algorithm, you are likely to need to run all of the three-dimensional simulations at a smaller system size, perhaps 50^3. If you have a fast computer, you may wish to run at a larger size, but make sure it is not tedious to watch.

Does it have a similar big jump, or is it continuous?

Below R_c we get a big jump; above R_c all avalanches are small compared to the system size. If the system size were large enough, we believe the fraction of spins flipped by the spanning avalanche at R_c would go to zero. The largest avalanche would nonetheless span the system—just like the percolation cluster at p_c spans the system but occupies zero volume in the limit of large systems.

The other avalanches form a nice power-law size distribution; let us measure it carefully. Do a set of 10 runs (*# Runs 10*) at $L = 100$ and $R = R_c = 2.16$.[3]

Watch the avalanches. Notice that sometimes the second-largest avalanche in the view (the largest being the 'background color') is sometimes pretty small; this is often because the cross-section we view missed it. Look at the avalanche size distribution. (You can watch it as it averages over simulations.) Print it out when the simulations finish. Notice that at R_c you find a pretty good power-law distribution (a straight line on the log–log plot). We denote this critical exponent $\bar{\tau} = \tau + \sigma\beta\delta$:

$$D(S, R_c) \sim S^{-\bar{\tau}} = S^{-(\tau+\sigma\beta\delta)}. \qquad (2)$$

(b) *From your plot, measure this exponent combination from your simulation. It should be close to two. Is your estimate larger or smaller than two?*

This power-law distribution is to magnets what the Gutenberg–Richter law (Fig. 12.3(b)) is to earthquakes. The power law stems naturally from the self-similarity.

We want to explore how the avalanche size distribution changes as we move above R_c. We will do a series of three or four runs at different values of R, and then graph the avalanche size distributions after various transformations.

Do a run at $R = 6$ and $R = 4$ with $L = 100$, and make sure your data files are properly output. Do runs at $R = 3$, $R = 2.5$, and $R = 2.16$ at $L = 200$.

(c) *Copy and edit your avalanche size distribution files, removing the data after the first bin with zero avalanches in it. Start up a graphics program, and plot the curves on a log–log plot; they should look like power laws for small S, and cut off exponentially at larger S. Enclose a copy of your plot.*

We expect the avalanche size distribution to have the scaling form

$$D(S, R) = S^{-(\tau+\sigma\beta\delta)}\mathcal{D}(S(R - R_c)^{1/\sigma}) \qquad (12.65)$$

sufficiently close to R_c. This reflects the similarity of the system to itself at a different set of parameters; a system at $2(R - R_c)$ has the same distribution as a system at $R - R_c$ except for an overall change A in probability and B in the size scale of the avalanches, so $D(S, R - R_c) \approx AD(BS, 2(R - R_c))$.

(d) *What are A and B in this equation for the scaling form given by eqn 12.65?*

At $R = 4$ and 6 we should expect substantial corrections! Let us see how well the collapse works anyhow.

(e) *Multiply the vertical axis of each curve by $S^{\tau+\sigma\beta\delta}$. This then should give four curves $\mathcal{D}(S(R - R_c)^{1/\sigma})$ which are (on a log–log plot) roughly the same shape, just shifted sideways horizontally (rescaled in S by the typical largest avalanche size, proportional to $1/(R - R_c)^{1/\sigma}$). Measure the peak of each curve. Make a table with columns R, S_{peak}, and $R - R_c$ (with $R_c \sim 2.16$). Do a log–log plot of $R - R_c$ versus S_{peak}, and estimate σ in the expected power law $S_{\mathrm{peak}} \sim (R - R_c)^{-1/\sigma}$.*

(f) *Do a scaling collapse: plot $S^{\tau+\sigma\beta\delta}D(S, R)$ versus $(R - R_c)^{1/\sigma} S$ for the avalanche size distributions with $R > R_c$. How well do they collapse onto a single curve?*

The collapses become compelling only near R_c, where you need very large systems to get good curves.

[3] If your machine is slow, do fewer. If your machine is fast, use a larger system. Make sure you do not run out of RAM, though (lots of noise from your hard disk swapping); if you do, shift to the *bits* algorithm if its available. *Bits* will use much less memory for large simulations, and will start up faster than *sorted list*, but it will take a long time searching for the last few spins. Both are much faster than the brute-force method.