Condensation in porous media: a fractal analysis of avalanches

Jake Skelton, supervised by Dr. Sergei Taraskin

16th May 2022

Abstract

The random-field Ising model is the archetypical system for modelling phase transitions in disordered systems, and it provided the first example of a disorder-induced critical point. More recently, analogous critical behaviour has been discovered in the porous lattice gas model of condensation in a disordered medium. It is an open question whether these two critical points are in the same universality class, as is the case for the conventional 3-dimensional Ising model and the liquid-gas transition. Here, we investigate this possibility by calculating the fractal dimensions of the so-called spanning avalanches that occur in the porous lattice gas transition; these dimensions are universal exponents. We estimate that the volumetric fractal dimension of one type of spanning avalanche, known as subcritical 3-dimensional, is 0.971 ± 0.003, in contradiction with previous estimates from the random-field Ising model, of 0.993 ± 0.007. There remain, however, doubts about the validity of this number.

1 Introduction

Perhaps the most significant consequence of Wilson's theory of the renormalisation group (RG) [1] is to put on firm ground previous notions that disparate physical systems can display *universal* behaviour at a phase transition. The RG approach, then, lies behind early ideas such as Landau mean-field theory [2] and the 'law of corresponding states' for gas thermodynamics [3]. In modern terminology, physical systems with identical RG flow dynamics at the critical point are said to lie in the same *universality class*. One of the most well-studied universality classes is that comprising three-dimensional systems with one-component fields; both the ferromagnetic phase transition of the $d = 3$ Ising model and the liquid-gas transition are members of this class.

Contemporaneously with the development of the RG, researchers aiming to model the disorder present in real systems invented the *random-field* Ising model (RFIM) [4]. In this lattice model, each site has its own static field drawn from a distribution; the fields are usually taken to be Gaussian-distributed in magnitude and uncorrelated with those on other sites. Much of the early interest in the RFIM stemmed from the fact that random fields of *any* strength are sufficient to prevent an ordered phase in all dimensions $d \leq 2$ [4, 5]; by contrast, the lower critical dimension of the conventional Ising model is $d_l = 1$.

More recently, Sethna and colleagues [6, 7] discovered that the RFIM undergoes a first-order phase transition at zero temperature which is qualitatively different depending on the variance of the random fields. For a small variance, i.e. low disorder, a magnetised system subject to an adiabatically-increasing external field reverses magnetisation direction in a single event, giving one large discontinuity in the magnetic hysteresis curve. On the other hand, a highly disordered system will track the applied field, and reverse magnetisation in a continuous fashion. At an intermediate variance, discontinuities of all possible sizes appear in the hysteresis curve, distributed according to a power-law [6]. Although a first-order transition, this phenomenon has been called a *disorder-induced* critical point [8], and the discontinuities, named *avalanches*, spurred further interest with their resemblance to the Barkhausen noise that is familiar in experimental ferromagnetic hysteresis curves [9].

Given the correspondence between the conventional Ising model and the liquid-gas transition, it is natural to ask whether this correspondence survives the introduction of disorder into both systems, and then, what is such a disordered fluid system? The leading answer to this question is the condensation of vapour in a highly porous random structure such as aerogel [10].

The object of this paper is to shed some more light on the possibility of a universality class shared by the disorder-induced phase transition in the RFIM, and the liquid-gas transition in porous media. Our approach is to analyse the geometry of avalanches in the latter transition, using a scheme which we call the *porous lattice gas model* (PLGM). Following this brief introduction, section 2 describes the PLGM, and discusses the theoretical ideas behind avalanches in greater depth. In section 3, we outline some of the details of how we simulate the dynamics of condensation. Section 4 is concerned with the methods used to characterise the fractal geometry of avalanches, then in section 5 we detail the results of this effort, alongside general observations regarding the avalanches. Finally, we discuss the implications of our findings in section 6 and offer concluding remarks in section 7.

2 Background

2.1 The model

Theoretical interest in modelling liquid-gas transitions in porous media was spurred by early experimental work [11]. De Gennes [12, 13] was the first to cast the problem as one of random fields, and Kierlik [14] was the first to propose the Hamiltonian which we employ, and refer to throughout this work as the porous lattice gas model:

$$\mathcal{H} = -w_{\mathrm{mf}} \sum_{\langle ij \rangle} \eta_i \eta_j \, \tau_i \tau_j \tag{1}$$

$$- w_{\mathrm{ff}} \sum_{\langle ij \rangle} \left(\tau_i \eta_i (1 - \eta_j) + \tau_j \eta_j (1 - \eta_i) \right) \tag{2}$$

where $\langle ij \rangle$ refers to the set of nearest-neighbour pairs, $\eta_i \in \{0, 1\}$ is the fluid occupancy of site i, and $\tau_i \in \{0, 1\}$ the matrix occupancy. Hence, the first sum in the Hamiltonian is over the matrix-fluid interactions, with w_{mf} the corresponding energy, and the second is over the attractive ($w_{\mathrm{ff}} > 1$) fluid-fluid interactions.

Any instance of the model is characterised by three thermodynamic variables. The *wettability* $y = w_{\mathrm{mf}}/w_{\mathrm{ff}}$ is the ratio of interaction energies. The *porosity* ϕ is the proportion of sites not occupied by matrix. Finally, the total number of fluid-occupied sites is not fixed, rather the model is treated via a grand canonical ensemble, and the *chemical potential* μ defines the relationship between fluid in the system and reservoir.

The standard mapping between the lattice gas and the Ising model can be generalised in this case to [14, 15]

$$\mathcal{H}_{\mathrm{Ising}} = -\frac{w_{\mathrm{ff}}}{4} \sum_{\langle ij \rangle} \eta_i \eta_j \, s_i s_j - \sum_i h_i s_i \tag{3}$$

with $s_i = 2\tau_i - 1$, and the random on-site field

$$h_i = \mu/2 + w_{\mathrm{mf}} \zeta_i + w_{\mathrm{ff}} (z/2 - \zeta_i) \tag{4}$$

where z is the coordination number of the underlying lattice, and ζ_i is the number of matrix sites bordering site i. For any realistic model of a porous lattice, ζ_i is a random quantity, and so the matrix gives the system *quenched disorder*.

Superficially, the above mapping looks to produce the RFIM, but the constant offset of h_i means this is not so [16]. The RFIM features zero-centred on-site fields with no site-site correlation [5], and consequently a symmetry in the Hamiltonian, which the above mapping fails to reproduce. It is for this reason that the correspondence between the PLGM and the RFIM, if it exists, is non-trivial.

2.2 Avalanches in the PLGM

It did not take long [17] for the characteristic features of the disorder-induced phase transition in the RFIM, as outlined in section 1, to be recognised in the first-order liquid-gas transition of the PLGM. The relevant order parameter, in this case, is the fluid concentration ρ rather than the magnetisation; the applied field is replaced by the chemical potential μ, and the disorder by the pair of parameters porosity ϕ and wettability y. A physical explanation for the dual behaviour either side of the critical disorder is straightforward for the PLGM. For a low-disorder, highly porous system, there is little preventing long-range correlation between sites; condensation of fluid at one site quickly cascades from nearest neighbours outward. For the highly-disordered system, the increased density of matrix impedes long-range correlation between sites: an avalanche can only spread to its immediate vicinity. From an energy perspective, the low-porosity system is like a spin glass [18, 19], its energy landscape consists of many metastable local minima. Smooth increase of the magnetisation as μ is increased corresponds to the location of the currently-occupied minimum changing; avalanches correspond to a discontinuous shift from one local minimum to a lower one, when the energy difference passes some threshold.

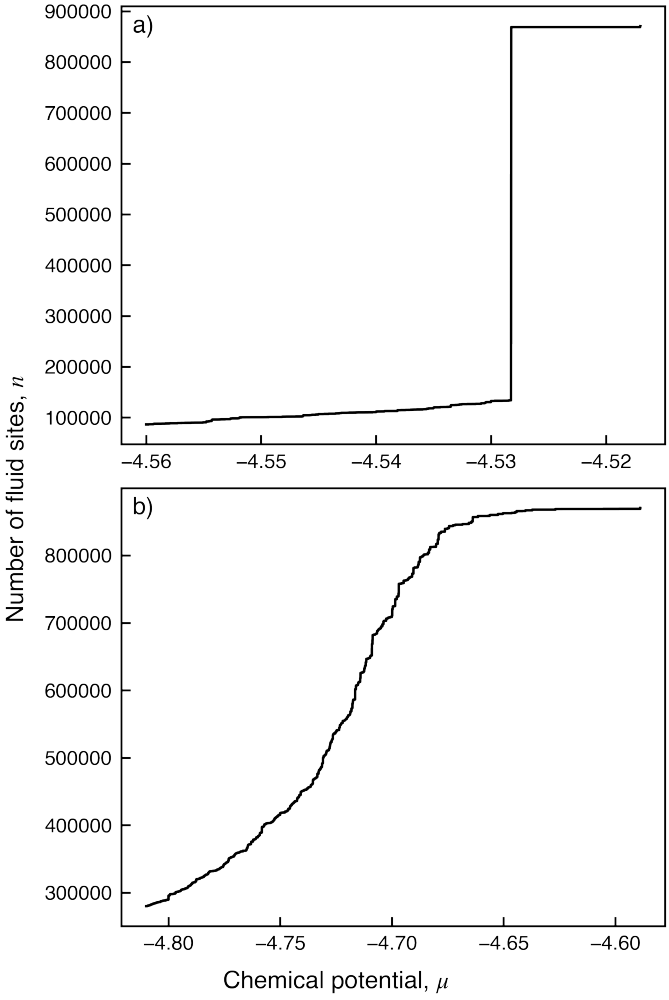


Fig. 1: Representative examples of isotherms a) below and b) above the critical disorder, in the porous lattice gas model. Note that the former is dominated by a single large avalanche, and the latter includes several small avalanches. In the thermodynamic limit, these behaviours tend to a single discontinuity, and a smooth curve.

The number of avalanches involved in a transition, and

their size, obey scaling relations near the critical point [7, 8], and are associated with a set of critical exponents. In this work, we are concerned with the fact that one way of combining the critical exponents yields the *fractal dimension* of the avalanches [7, 20]. A brief introduction to fractal geometry, along several authoritative references, is given in section 2.4. Because it can be written in terms of other critical exponents which appear in a renormalisation group analysis [8], d_f is a bona-fide critical exponent that is characteristic of the universality class to which the RFIM belongs.

2.3 Spanning avalanches

In a simulation of a system with a finite number of sites N, with periodic boundary conditions, there is a distinction to be made between avalanches which reach from one side of the system to the other, and those which can be comfortably contained within the system size. The former are referred to as *spanning avalanches*. In the thermodynamic limit $N \to \infty$, spanning avalanches represent an event where an infinite number of spins flip, whereas non-spanning avalanches remain finite as $N \to \infty$ [8]. In fact, in the thermodynamic limit, there is a single infinite avalanche below the critical disorder, $R < R_c$, only finite avalanches above R_c, and an infinitude of infinite avalanches at the critical disorder [7, 8, 21].

In this work, we refer to an avalanche that spans one pair of opposite faces of the system, as a '1-dimensional' spanning avalanche; an avalanche that spans two pairs of opposing faces is '2-dimensional', etc. Pérez-Reche and Vives [21, 22] have proposed that the single subcritical infinite avalanche that appears in the thermodynamic limit is obtained only from the limit of a subset of the 3-dimensional spanning avalanches encountered in finite systems, which they call 'subcritical 3-dimensional spanning avalanches'; in this paper, they are denoted $3^<$. The remainder of the spanning avalanches, which include 1-, 2-, and 3-dimensional types, produce the infinitude of avalanches at the critical point as $N \to \infty$. They are referred to as 'critical' spanning avalanches and here denoted 1*, 2*, and 3*, respectively. The above classifications of avalanche are the basis upon which results are divided throughout the rest of this paper, so they are given for reference in table 1.

In other words, the finite-size scaling (FSS) analysis in [21, 22] implies that the distribution of 1*, 2*, and 3* avalanches tends to $\delta(R - R_c)$ as $N \to \infty$, whereas that of $3^<$ tends to the step function $\Theta(R - R_c)$. A plot illustrating this is reproduced from [22] in fig. 2. To the author's knowledge, no analogous FSS procedure distinguishing the distribution of avalanche types has been rigorously carried out for the PLGM, and the difficulties involved have been pointed out by previous authors [23]. In this work therefore, we adopt Peréz-Reche's classification of avalanches as a working hypothesis, and it will be found that the qualitative features of avalanches in the PLGM do indeed conform to this scheme.

1*	1-dimensional spanning avalanches
2*	2-dimensional spanning avalanches
3*	3-dimensional, critical spanning avalanches
$3^<$	3-dimensional, subcritical spanning avalanches

Table 1: Abbreviations for various types of spanning avalanche, following [22].

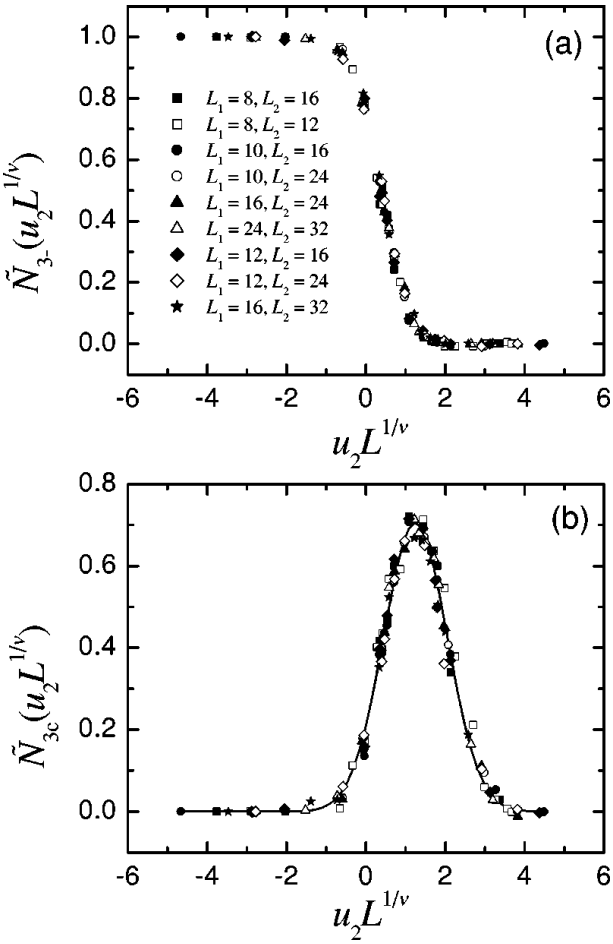


Fig. 2: Scaling collapse to the hypothesised distributions for the number of a) $3^<$ and b) 3* spanning avalanches (in our notation). For the purposes of this paper, the horizontal axis represents a (scaled) distance from the critical disorder $u = 0$, and the vertical axis the scaling function for the distribution of each type of avalanche. Reproduced from [22].

2.4 Fractals

The concept of a fractal was devised almost entirely by Mandelbrot, building on some disparate work by early 20th century mathematicians [24]. The two loose definitions of a fractal, which are equivalent when expressed more formally are i) a fractal is an object with structure, or detail on *all lengthscales*; ii) a fractal is an object with a scaling dimension greater than that possessed by conventional objects with the same topology, but smaller than that of the embedding space [25]. This *fractal dimension* is frequently non-integral.

The resemblance of fractals to real objects was always an impetus for their development [24], and their utility for characterising physical systems was realised early by

researchers [26]. Fractal ideas are particularly prominent in the computational literature, for example in studies of percolation [27], diffusion-limited aggregation [28], and chaos [29]. Additionally, the divergence of the correlation length at a critical point leads to statistically self-similar structures which are well-suited to fractal analysis [30].

Formally, there are many different fractal dimensions, but they coincide in many cases. The most important is the *Hausdorff dimension* [31, 32]. In this work, we use the *mass dimension*, a precise definition of which requires, like many concepts in fractal geometry, a certain amount of set theory. Loosely, however, if $\mathcal{F}$ is a (potentially fractal) set, $\{\mathcal{U}_j\}$ is a cover of $\mathcal{F}$, and μ is a measure (subject to some constraints), then the mass dimension relates the measure $\mu(\mathcal{U}_j)$ of any $\mathcal{U}_j$ to its diameter (in the set-theoretic sense). Crucially for our purposes, while this definition requires there to be a metric on $\mathcal{U}_j$, there is no need for $\mathcal{F}$ to be a subset of Euclidean space $\mathbb{R}^n$. For a proper exposition, the reader is urged to consult [33] and the references therein.

3 Simulation

The results reported in this work were obtained by simulating the PLGM on $L \times L \times L$ body-centred cubic (BCC) lattices of linear sizes L = 20, 30, 40, 50, 60, and 80. Previous analysis by Dr Taraskin has suggested the point $\phi = 85\%$, $y = 1.03$ in the porosity-wettability plane is close to the critical line, and these values are used throughout this work.

3.1 Creating the matrix

The aerogel matrix, which provides the quenched disorder, was realised using a diffusion-limited cluster aggregation (DLCA) [34, 35] algorithm with an enforced, user-chosen porosity. The code for this algorithm was written by S. Niblett and the procedure detailed in a previous report*[36] and we direct the reader there for further information. For the purposes of this work, it suffices to say that the DLCA algorithm creates a random matrix configuration each time it is executed. Then, we calculate one condensation event (upward branch of the isotherm) per configuration; repeating with a greater number of random initialisations of the matrix leads naturally to better statistics. In this work for L = 20, 30, 40, 50, 60, and 80 systems we have performed simulations using 6000, 20 000, 25 000, 18 000, 18 000 and 3000 random matrix configurations, respectively.

3.2 Calculating condensation dynamics

For a given chemical potential, we then solve for the ground state of the system using the long-established (though perhaps lesser-known) mapping between ground state determination for certain lattice Hamiltonians, and the *minimum-cut* optimisation problem from the theory of network flows [38, 39]. Briefly, this is the problem of bi-partitioning a weighted graph with a 'cut' made of contiguous edges such that a) two particular nodes end up on opposite sides of the partition, and b) the weights of the edges used in the cut sum to a minimal value. For a d-dimensional lattice of two-state sites, the minimum-cut is a boundary in $d+1$ dimensional space, and its intersection with d-space gives the ground-state domain walls. In practice, the minimum-cut can be found using one of several algorithms which invoke the *maximum-flow/minimum-cut theorem* [40], a key result of network flow theory; we use an algorithm due to Boykov and Kolmogorov [41]. The specifics of applying the max-flow/min-cut theorem to the PLGM were implemented by R. Cepitis and detailed in an earlier work [37]. For more information on the general application of network flow theory to ground-state determination, the reader is referred to the excellent review by Alava et al. [39].

In order to build up the isotherm and find the avalanches, which manifest as discontinuities, one could simply sample a suitable interval in μ with a fixed step-length $\delta\mu$. However, this is wasteful when evaluating at several values of μ that all correspond to a plateau in ρ, for example at the start and end of the curve. Instead, we follow [23] and use a form of binary search that recursively halves an interval in μ and evaluates ρ at the start-, end- and midpoints. A half-interval is discarded from the recursion if $\Delta\rho = 0$ across it, and the recursion finishes once intervals reach a pre-specified step length $\delta\mu$. In this way, the isotherm is only finely sampled where necessary.

If one solves for the ground state of the system immediately before and after a discontinuity in ρ, then finds the differential in fluid occupation of each site in the system, the result is the spatial representation of the avalanche. Representative examples of such avalanche shapes are shown in figs. 4 and 5.

3.3 Classifying avalanches

In order to determine, in practice, whether a 3-dimensional spanning avalanche is critical or subcritical, we follow the method of Pérez-Reche in [22]. In the RFIM it was found, through combining other critical exponents, that $3^<$ avalanches have a fractal dimension close to the spatial dimension $d = 3$, and thus occupy a large fraction of the system, whereas the dimension for 3* avalanches is smaller. This makes intuitive sense if 3* avalanches are to coexist with other spanning avalanches at the critical point, in the thermodynamic limit. This fact suggested the following classification scheme, used in [22]: if any other spanning avalanches occur on the same isotherm as a 3D spanning avalanche, that avalanche is critical, 3*, otherwise, if it is alone, it is $3^<$. The extent to which this scheme is justified for our studies of the PLGM will be discussed in section 6.2

*References [36] and [37] are previous student project reports and are uploaded as material supplementary to this report.

4 Fractal analysis

To estimate the fractal dimension of an avalanche starting from its spatial representation, we use the well-established *sandbox algorithm* [26]. The fortran code used to implement it is provided in appendix B, and here we shall outline some of the practicalities of applying the algorithm.

4.1 The sandbox algorithm

The sandbox method for estimating fractal dimension is based simply on the definition of the fractal scaling. That is, for an object contained within a d-volume V, the mass M (in a generalised sense) in a smaller volume v obeys

$$\begin{aligned} M(v) &= \left(\frac{v}{V}\right)^{d_f/d} M(V) \\ &\equiv M_0 v^{d_v} \end{aligned} \tag{5}$$

where d_f is the fractal dimension of the object, M_0 is a constant and $d_v = d_f/d$ is a quantity we refer to as the *volumetric fractal dimension*. The above expression is a slight rearrangement of that seen typically [24, 26] because applying the sandbox method to avalanches in the PLGM is complicated significantly by the presence of the matrix. The approach we adopt in this work, and the key deviation from the sandbox method, as usually applied, is as follows.

0. Consider a $L \times L \times L$ system with a BCC lattice, so that there are L^3 unit cells, each with 2 sites. Define $\mathcal{A}$ as the set of all fluid sites $\vec{x} \in \mathcal{A}$ that comprise the avalanche, and define $\mathcal{L} = 1, 2, ..., L$ as the set of all integers between 1 and L.
1. Select a site $\vec{x}$ from $\mathcal{A}$.
2. Select a value l from $\mathcal{L}$. Draw a $l \times l \times l$ 'bounding-box' centred on $\vec{x}$, which may wrap around the periodic boundaries.
3. Calculate the number of fluid sites $n(\vec{x}; l)$, and the number of non-matrix, 'pore' sites $v(\vec{x}; l)$ in the bounding-box.
4. Repeat steps 2-3 for all box sizes $l \in \mathcal{L}$.
5. Repeat steps 1-4 for all avalanche sites $\vec{x} \in \mathcal{A}$.
6. Let $\mathcal{X}_{v'}$ be the set of all $(\vec{x}, l)$ pairs such that $v(\vec{x}; l) = v'$.
7. For every encountered 'sandbox size' v, output the average fluid occupation,

$$\bar{n}(v) \equiv \frac{1}{|\mathcal{X}_v|} \sum_{(\vec{x},l)\in\mathcal{X}_v} n(\vec{x}; l). \tag{6}$$

Using eq. (5), the expectation is then that $\bar{n}(v)$ scales as v^{d_v}. The key differences between this method and the usual incarnation of the sandbox method is that we investigate the scaling with a volumetric quantity v rather than a linear quantity, and these volumes are not pre-defined but are discovered at 'run-time'. The reason for the deviation is that the pore space in which an avalanche is embedded is highly non-trivial, and we must account for the fact that matrix sites are unavailable for the growth of the avalanche.

In practice, the objects central to the algorithm are two $L \times L \times L$ arrays containing, respectively, the fluid occupation, and the matrix occupation of each unit cell (with each element being 0, 1, or 2 due to the BCC lattice). The nested loop structure of the algorithm means step 3, which performs a sum over a $l \times l \times l$ slice of both arrays, is visited a great many times. It was realised that the static nature of the occupation arrays can be leveraged to accelerate these sums. The technique used relies on the *integral image* construction from the image processing literature [42–44]. A detailed description is given in appendix A, but the concept is totally analogous to the use of a cumulative distribution function to save the effort of integrating a probability density function. In the end, a sum of $2l^3$ integers per iteration was reduced to seven integer additions per iteration, plus the constant-time evaluation of the integral image itself. Altogether, this decreases the scaling of execution time with system size from $O(L^{4+d_f})$ to $O(L^3)$.

4.2 Scaling collapse

We will observe later in this work that finite-size effects impinge upon the expected power-law fractal scaling as v approaches the system size L^3. Fitting a power-law, and especially a truncated one, is fraught with difficulties in any case [45, 46], and so the choice was made to determine the asymptotic scaling behaviour using *scaling collapse*. This is a well-established technique [47], and so we shall explain it only briefly. Equation (5) is truly only valid in the thermodynamic limit of an infinite fractal. Away from this limit, it must be generalised to the finite-size scaling form [21, 48, 49]

$$n(v) = V^{d_v} f_n(v/V) \tag{7}$$

where f_n is a universal scaling function. This means that eq. (7) holds for a simulated PLGM of any size and any pair of y and ϕ that tune the model to the critical point. In particular, if one plots numerical results for $n(v)$ as $V^{-d_v} n(v)$ versus v/V, then the various datasets should all 'collapse' to lie on a single curve. Figures 7 and 8 should illustrate this idea more clearly.

In order to obtain the parameter-set that yields best collapse (in a quantitative sense), various heuristic cost-functions have been developed [50–52], which all have the formidable task of evaluating the mutual displacement of scaling curves that have been only discretely sampled, at locations and sample-densities that do not coincide in general. In this work, we use the cost-function due to Houdayer and Hartmann [52], which is a modification of that due to Kawashima and Ito [50]. For more detail the reader should consult [52], but the general idea is as follows. For every possible point P from every dataset, compute the

residual between P and a straight line segment fit to the immediate neighbours of P from all other datasets. The residual is scaled by the error in P and in the linear fit, and then summed together with all other such scaled residuals (controlling for the number of terms) to form a scalar 'quality' of collapse, Q. Minimising Q will therefore find the parameter-set that optimises collapse, but, by analogy with the χ^2 test [53], $Q \ll 1$ is suspect and implies oversized errors.

In this work, we apply $Q[(n, v); d_v]$ to a brute-force search over a pre-defined vector of d_v values. The d_v that minimises Q (subject to the resolution of the initial vector) is taken as our estimate of the scaling exponent, and the 'half-width at half-minimum' (HWHM) is used as a heuristic for the error on the estimate. The code for this procedure was provided by Dr Taraskin.

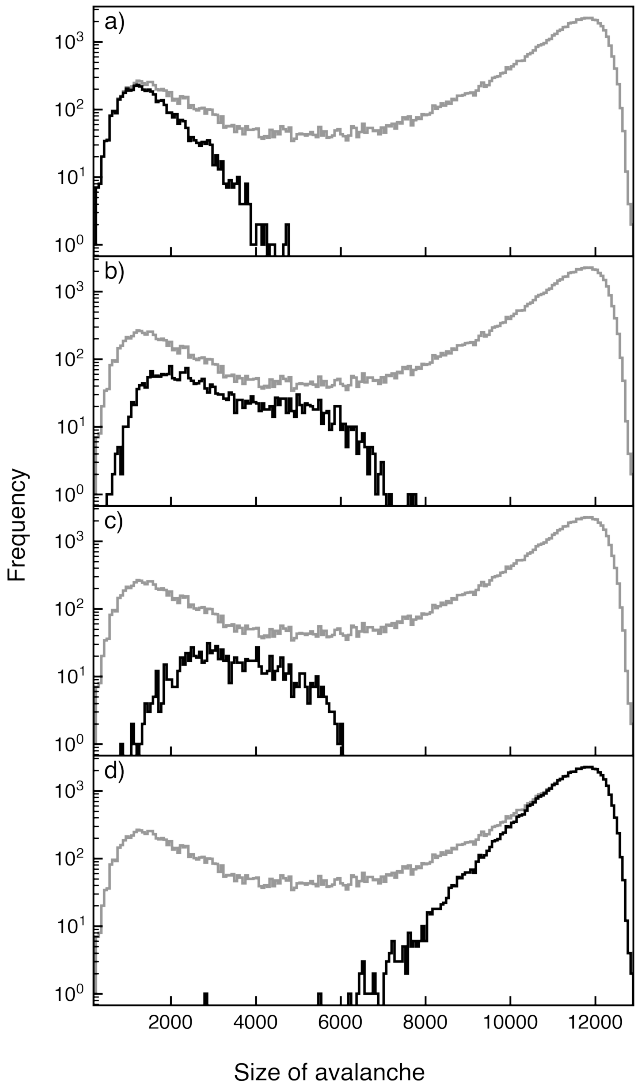


Fig. 3: Distribution of sizes of 61 214 spanning avalanches recorded in 60 000 realisations of a $L = 20$ system. Each plot has superimposed (in grey) the distribution of all types of spanning avalanche, alongside a) 1*, b) 2*, c) 3*, d) 3$^<$. Note the logarithmic scale of the vertical axis. The very largest recorded avalanches for types 1*, 2*, and 9* are 4750, 7764, and 6033, respectively.

5 Results

5.1 Observations

Computing the system state using max-flow/min-cut techniques identifies all sites involved in an avalanche, and allows us to visualise the avalanches themselves. Figures 4 and 5 show representative examples of 1* and 2*, and type 3* and 9* avalanches, respectively. Their fractal nature is intuitively clear. It is also evident that 3$^<$ avalanches are qualitatively different from other spanning avalanches: informally, the former are defined more by those sites that are not occupied, with the latter defined by than those that are.

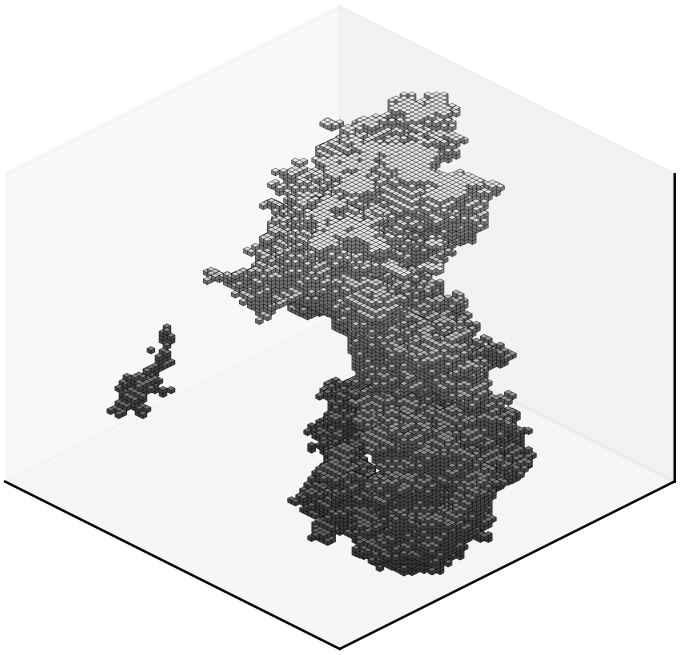
5.2 Distribution of avalanches

The distribution of spanning avalanche sizes is distinctly bimodal, as illustrated in fig. 3 for $L = 20$ systems. Classification of the avalanches allows the distribution to be decomposed: critical spanning avalanches, predominantly 1* and 2* are responsible for the peak at smaller sizes, and 3$^<$ are solely responsible for it at larger sizes. This is not a totally trivial finding; although it is more likely for a smaller object to intersect fewer system boundaries, we do not, for example, find a single 1* or 2* avalanche that contributes to the peak at large sizes, among 60 000 $L = 20$ systems.

5.3 Scaling collapse

As remarked upon by previous authors dealing with the RFIM [21], the variability among avalanches is enormous. Figure 6 shows the scaling of fluid volume $n(v)$ versus pore-space 'sandbox volume' v for the $L = 20$ dataset. The point clouds involved overlap to a significant extent in n, whereas the individual contributions from each bounding box are distinct in v. Evidently, the DLCA procedure ensures that, the fraction of any given volume occupied by matrix is governed by some distribution, centred around $1 - \phi$; for $\phi = 85\%$, this distribution is narrow.

In order to make the data amenable to scaling collapse, the data corresponding to each of the L bounding boxes were averaged in n and v. This gave us seven datasets (one for each simulated system size) with which to perform scaling collapse, for each of the four types of spanning avalanche. The results of the procedure outlined in section 4.2 are shown in figs. 7 and 8. With the notable exception of 1* and 2* avalanches in $L = 20$ systems, the collapse is very good. On the logarithmic plots, finite-size effects, which manifest as a 'knee' in the curve, are particularly evident for 1* and 2*. The linear plot of scaling for 3$^<$ avalanches is nearly straight; a linear relationship between $n(v)$ and v would imply $d_v = 1$, and a compact, rather than fractal, object.



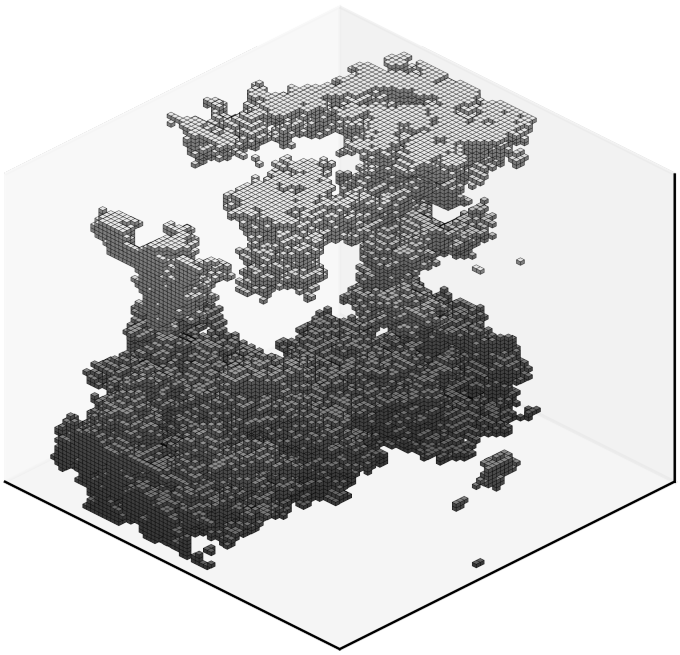
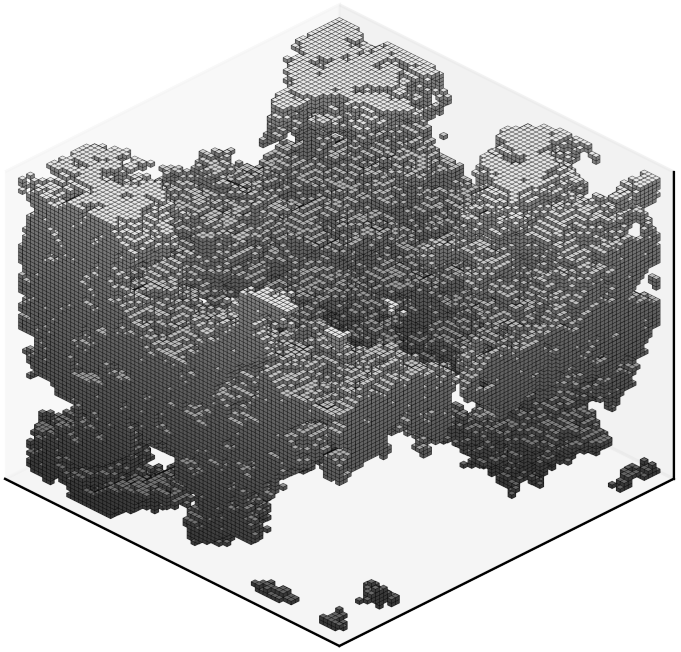


Fig. 4: Visualisations of representative 1* (above) and 2* (below) avalanches in a $L = 80$ system. Each voxel (3D pixel) represents one (singly or doubly) occupied unit cell in the BCC lattice. Unit cells are coloured on a linear scale from dark grey to white based upon their z-coordinate, to aid the eye. What appear to be stranded 'islands' of fluid are in fact connected to the main edifice under periodic boundary conditions.



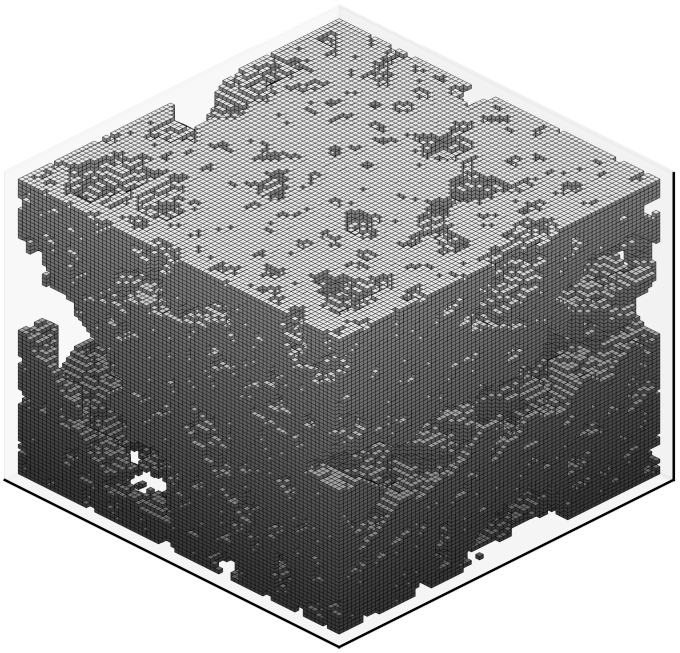


Fig. 5: Visualisations of representative examples of the two 3-dimensional spanning avalanches, 3* (above) and $3^<$ (below), here in a $L = 80$ system. Each voxel (3-dimensional pixel) represents one (singly or doubly) occupied unit cell in the BCC lattice. Unit cells are coloured on a linear scale from dark grey to white based upon their z-coordinate, to aid the eye. The 3* avalanche is quite obviously more diffuse than the $3^<$. It should be noted that this figure does not indicate which of those non-fluid sites are free, and which are occupied by matrix.

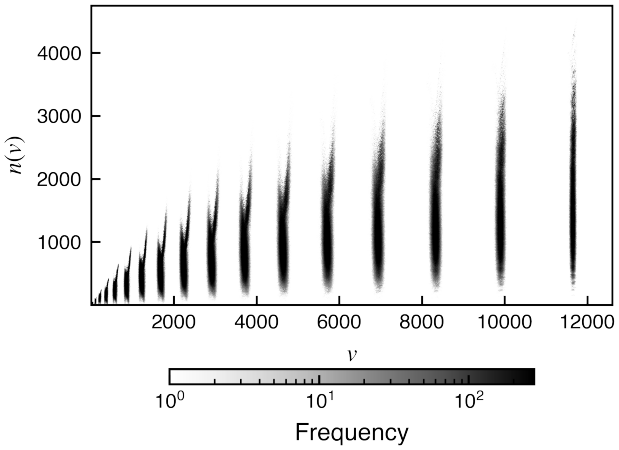


Fig. 6: Raw volume scaling output of the sandbox procedure applied to 4570 1* avalanches from $L = 20$ systems. The plot is a histogram with 2000×2000 linearly-spaced bins, the occupation of each bin is indicated using a logarithmic colour scale such that all bins exceeding the mean occupation by more than five standard deviations are coloured black. Note the broad distribution of fluid occupations n of each sandbox, and the fact that the distributions stemming from each 'bounding-box' are distinguishable. The distribution corresponding to the final bounding box, of size 20^3, is not shown in order to preserve the dynamic range of the colouring.

The numerical results of the scaling collapse are collected in table 2. Also included are the fractal dimensions for the four types of avalanche in the RFIM, reported by Pérez-Reche and Vives in [21, 22], re-expressed as volumetric, rather than linear, scaling exponents. The implications of these findings will be discussed in depth in section 6, and here we restrict the analysis to the following remarks. To within error, estimates of d_v for avalanches of type 1* and 2* are consistent both with each other, and with the corresponding exponents in the RFIM. The 3-dimensional avalanches have estimated exponents consistent with each other, which is not the case for the RFIM, and neither 3* nor 3$^<$ have d_v estimates in agreement with the RFIM values.

Aval. type	Dimension, d_v	Quality	RFIM value [22]
1*	0.95 ± 0.02	9.61	0.93 ± 0.02
2*	0.93 ± 0.02	13.38	0.93 ± 0.02
3*	0.96 ± 0.01	1.03	0.93 ± 0.02
3$^<$	0.971 ± 0.003	51.32	0.993 ± 0.007

Table 2: Results of the scaling collapse for volumetric fractal scaling, along with the 'quality' of the collapse as defined in section 4.2, and the corresponding fractal dimensions for the RFIM, from [22]. The error in d_v is simply the half-width at half-minimum of the collapse cost function $Q[d_v]$. The values for the RFIM reported in [22] are for scaling with linear resolution, so they are presented here divided by the spatial dimension, 3.

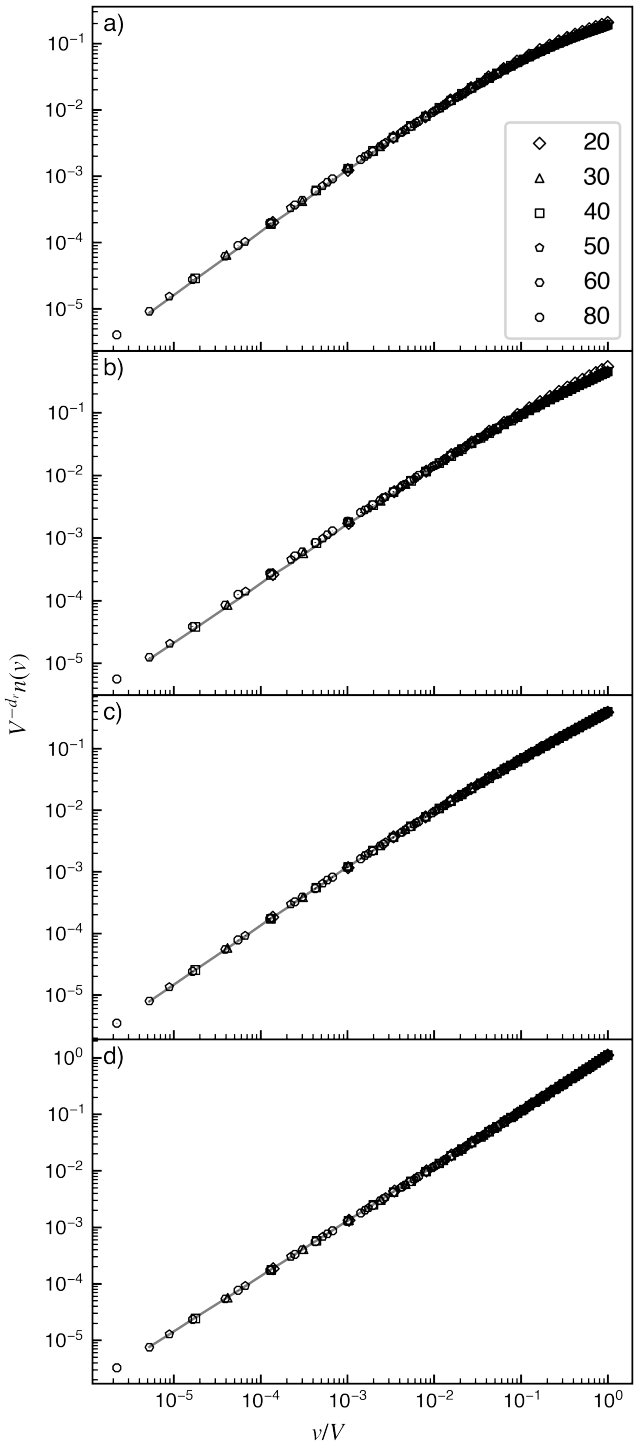


Fig. 7: Scaling collapse for fluid occupation $n(v)$ of a porous sandbox versus sandbox volume v, using eq. (7). The plots, on a logarithmic scale are for a) 1*, b) 2*, c) 3*, and d) 3$^<$ spanning avalanches. The six datasets for the six system sizes are indicated with the marker shape: $L = 20$ are represented by diamonds, the remainder by the corresponding L-sided polygon. The error bars on each datum are shown for completeness.

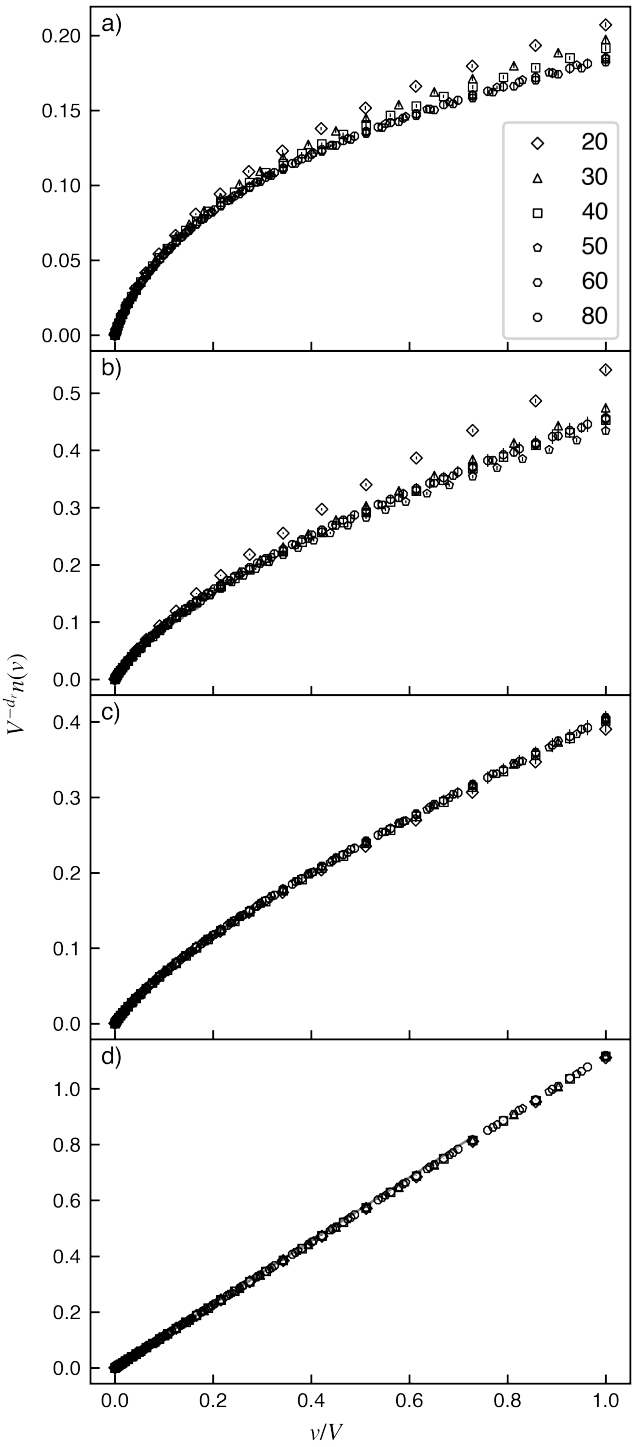


Fig. 8: Scaling collapse for fluid occupation $n(v)$ of a porous sandbox versus sandbox volume v, using eq. (7). The plots are for a) 1*, b) 2*, c) 3*, and d) $3^<$ spanning avalanches. In contrast to fig. 7, linear axes are used here. The six datasets for the six system sizes are indicated with the marker shape: $L = 20$ are represented by diamonds, the remainder by the corresponding L-sided polygon. The error bars on each datum are shown for completeness.

6 Discussion

The principle question driving this work is this: do the disorder-induced phase transitions of the random-field Ising model and the porous lattice gas model share a universality class? Identical avalanche fractal exponents are a necessary condition for the renormalisation group dynamics to be the same at the critical point, though not a sufficient condition because the other critical exponents offer further degrees of freedom [1].

As it is, our situation is less clear cut than an affirmation or a refutation of universality for the two models. The estimates for exponents $d_v(1*)$ and $d_v(2*)$ appear consistent with those for the RFIM, but the errors involved, for both models, are too large to make any firm statements. Concerning the 3-dimensional spanning avalanches, if i) the hypothesis that there are two fundamentally different types of 3-dimensional avalanche applies to the PLGM, and ii) if the method used in this work to distinguish them is correct, then the precision of the estimates of $d_v(3*)$ and $d_v(3^<)$, for both models, is sufficient to preclude a shared universality class for the disorder-induced critical points of the PLGM and the RFIM. However, if one or both of the conditions, (i) and (ii), are not met, then the exponents calculated for the PLGM are invalid and no firm conclusion can be drawn regarding universality. In the author's view, both conditions require further scrutiny.

6.1 Uncertainty

Much of the advice regarding the use of the sandbox method in the literature [26] suggests the use of as many sites in the fractal object as possible as centres for the drawing of sandboxes (or bounding-boxes, in our modified approach). The motivation is to collect as large a sample as possible for good statistics. There is perhaps a balance to be struck here. We found that using every fluid-occupied site in the avalanche led to significant non-independence of the data; the largest bounding-boxes drawn from one site overlap with those of almost every other site in the system. This led to the error-on-the-mean for the averaging described in section 5.3 being unphysically small if calculated using standard techniques assuming independent data. It was chosen to treat all the, say, N records for a given sandbox size v coming from a single matrix configuration as one datum, rather than N data, to overcome the problem. A more rigorous technique, perhaps using a random sample of origin sites per avalanche, should be sought for future analysis.

Besides this, a future direction to obtain greater precision would be the inclusion of not only more system realisations, but also larger systems, as these are less susceptible to the finite-size effects that so necessitate the scaling collapse technique. The deviation of the $L = 20$ dataset in fig. 8 is perhaps in support of this.

6.2 Dichotomy of 3-dimensional avalanches

Pérez-Reche and Vives [22] presented strong evidence for their hypothesis of two varieties of 3-dimensional spanning avalanche using a FSS analysis. Fig. 10 from [22], depicting a scaling collapse of data to their hypothesised scaling form is reproduced in fig. 2. The dual sigmoid, and unimodal behaviour expected is replicated in the data using only exponents extracted from fitting 1* and 2* avalanche statistics. Further, the authors were able to estimate the fractal dimensions of 3* and 3$^<$ avalanches by combining other scaling exponents, before conducting more traditional fractal scaling techniques.

In contrast, no such scaling analysis exists in the PLGM literature, to the author's knowledge. Without such an a priori indication of the need to distinguish two sets of 3-dimensional avalanches, the method outlined in section 3.3 is discomfortingly close to a tautology; in [21], 3$^<$ avalanches are posited to be large to the point of exclusivity based only on their known, larger fractal dimension. As it happens, our $d_v(3*)$ estimate is not precise enough to support such an a priori hypothesis that it differs from $d_v(3^<)$.

7 Conclusion

To conclude, we have estimated the fractal scaling exponents of spanning avalanches in the porous lattice gas model and compared these values to the best estimates for avalanches in the random-field Ising model. Doubts regarding whether it is valid to apply classification schemes developed for RFIM avalanches to those in the PLGM prevent us from concluding firmly that the two sets of fractal dimensions are, or are not consistent.

Further work is thus required to interrogate the qualitative subsets of spanning avalanches in the PLGM and, in particular, how the distribution of such subsets will behave in the thermodynamic limit. An analytical treatment with RG methods would be the most convincing approach.

However, should the hypothesis of a dichotomy of 3-dimensional avalanches be validated, and if the method to distinguish them used in this work is appropriate, then our scaling exponent for subcritical 3-dimensional spanning avalanches, 0.971 ± 0.003, appears incompatible with that from the RFIM, 0.993 ± 0.007. In the absence of error, this would rule out a universality class shared by the RFIM and the PLGM.

Word count = 4815

A 3-dimensional integral images

As remarked in section 4, the relationship between an array (or image in the historic terminology) and an integral image (II) is entirely analogous to that between a probability density function and a cumulative distribution function [42]. Each element of the II correspond to the sum over a particular region of the array, so that performing arithmetic with elements of the II is equivalent to summing a subset of the elements of the array. If one needs to perform several such sums over the array, then there comes a point where it is more efficient to first compute an II, then use it to do the arithmetic. As the name indicates, integral images were originally devised for 2-dimensional arrays, but they can be generalised to n-dimensions [44]. We shall restrict this exposition to three dimensions.

Firstly, we define an II and show how it can be used to compute the sum over any cuboid section of a 3-dimensional array. Then, we give an efficient method for creating an II using one pass over the original array.

Let A be a rank-3 array of shape (n_x, n_y, n_z); let I be the corresponding integral image, which is one element larger than A in each direction, $\text{shape}(I) = (n_x + 1, n_y + 1, n_z + 1)$. I is defined such that

$$I_{ijk} = \sum_{i'=1}^{i} \sum_{j'=1}^{j} \sum_{k'=1}^{k} A_{i'j'k'} \quad \text{for non-zero } i, j, k \tag{8}$$

and $I_{ijk} = 0$ if any of i, j, k are zero. Note that, under this definition, A is 1-referenced, and I is 0-referenced. This subtlety has a justification.

This then implies that the sum over a $m_x \times m_y \times m_z$ slice of A is

$$\begin{aligned} &\sum_{i'=i}^{i+m_x} \sum_{j'=j}^{j+m_y} \sum_{k'=k}^{k+m_z} A_{i'j'k'} = \\ &+ I_{i+m_x,j+m_y,k+m_z} \\ &- I_{i+m_x,j+m_y,k} - I_{i,j+m_y,k+m_z} - I_{i+m_x,j,k+m_z} \\ &+ I_{i+m_x,j,k} + I_{i,j+m_y,k} + I_{i,j,k+m_z} \\ &- I_{ijk}. \end{aligned} \tag{9}$$

Finally, the procedure we use to generate I is

$$\begin{aligned} I_{i+1,j+1,k+1} = &+ A_{i+1,j+1,k+1} \\ &+ I_{i+1,j+1,k} + I_{i,j+1,k+1} + I_{i+1,j,k+1} \\ &- I_{i+1,j,k} - I_{i,j+1,k} - I_{i,j,k+1} \\ &+ I_{ijk}. \end{aligned} \tag{10}$$

In fact, eq. (10) is not the conventional expression; it requires slightly more additions but is more memory-efficient and, we feel, easier to intuit because of its similarity to eq. (9). The earlier insistence that I be larger than A and that $I_{0jk} = I_{i0k} = I_{ij0} = 0$ is not strictly necessary, but it ensures eq. (10) is unambiguous for all i, j, k.

Except where specific reference is made to the work of others, this work is original and has not been already submitted either wholly or in part to satisfy any degree requirement at this or any other university.

References

1. Wilson, K. G. & Kogut, J. The Renormalization Group and the Epsilon Expansion. *Physics Reports* **12,** 75–199. ISSN: 0370-1573. doi:10.1016/0370-1573(74)90023-4 (1st Aug. 1974).
2. Landau, L. D. On the Theory of Phase Transitions. I. *Phys. Z. Sowjet.* **11,** 26 (1937).
3. De Boer, J. Quantum Theory of Condensed Permanent Gases I the Law of Corresponding States. *Physica* **14,** 139–148. ISSN: 0031-8914. doi:10.1016/0031-8914(48)90032-9 (1st Apr. 1948).
4. Imry, Y. & Ma, S.-k. Random-Field Instability of the Ordered State of Continuous Symmetry. *Physical Review Letters* **35,** 1399–1401. doi:10.1103/PhysRevLett.35.1399 (24th Nov. 1975).
5. Nattermann, T. & Villain, J. Random-field Ising Systems: A Survey of Current Theoretical Views. *Phase Transitions* **11,** 5–51. ISSN: 0141-1594. doi:10.1080/01411598808245480 (1st Jan. 1988).
6. Sethna, J. P. *et al.* Hysteresis and Hierarchies: Dynamics of Disorder-Driven First-Order Phase Transformations. *Physical Review Letters* **70,** 3347–3350. doi:10.1103/PhysRevLett.70.3347 (24th May 1993).
7. Dahmen, K. & Sethna, J. P. Hysteresis, Avalanches, and Disorder-Induced Critical Scaling: A Renormalization-Group Approach. *Physical Review B* **53,** 14872. ISSN: 1550235X. doi:10.1103/PhysRevB.53.14872. pmid: 9983282 (1st June 1996).
8. Perković, O., Dahmen, K. A. & Sethna, J. P. Disorder-Induced Critical Phenomena in Hysteresis: Numerical Scaling in Three and Higher Dimensions. *Physical Review B* **59,** 6106. ISSN: 1550235X. doi:10.1103/PhysRevB.59.6106. arXiv: cond-mat/9807336 (1st Mar. 1999).
9. Barkhausen, H. Zwei Mit Hilfe Der Neuen Verstärker Entdeckte Erscheinungen. *Phys. Z* **20,** 401–403 (1919).
10. Kistler, S. S. Coherent Expanded-Aerogels. *The Journal of Physical Chemistry* **36,** 52–64. ISSN: 0092-7325. doi:10.1021/j150331a003 (1st Jan. 1932).
11. Wong, A. P. Y. & Chan, M. H. W. Liquid-Vapor Critical Point of He4 in Aerogel. *Physical Review Letters* **65,** 2567–2570. ISSN: 0031-9007. doi:10.1103/PhysRevLett.65.2567 (12th Nov. 1990).
12. Brochard, F. & de Gennes, P. G. Phase Transitions of Binary Mixtures in Random Media. *Journal de Physique Lettres* **44,** 785–791. ISSN: 0302-072X, 2777-3434. doi:10.1051/jphyslet:019830044018078500 (1st Sept. 1983).
13. De Gennes, P. G. Liquid-Liquid Demixing inside a Rigid Network. Qualitative Features. *The Journal of Physical Chemistry* **88,** 6469–6472. ISSN: 0022-3654. doi:10.1021/j150670a004 (1st Dec. 1984).
14. Kierlik, E., Rosinberg, M. L., Tarjus, G. & Pitard, E. Mean-Spherical Approximation for a Lattice Model of a Fluid in a Disordered Matrix. *Molecular Physics* **95,** 341–351. ISSN: 0026-8976. doi:10.1080/00268979809483166 (10th Oct. 1998).
15. De, S., Shapir, Y., Chimowitz, E. H. & Kumaran, V. Critical Behavior in Quenched Random Structures: Mean-field Lattice-Gas Approach. *AIChE Journal* **47,** 463–473. ISSN: 1547-5905. doi:10.1002/aic.690470223 (2001).
16. Maritan, A. *et al.* Ordering and Phase Transitions in Random-Field Ising Systems. *Physical Review Letters* **67,** 1821–1824. doi:10.1103/PhysRevLett.67.1821 (30th Sept. 1991).
17. Detcheverry, F., Kierlik, E., Rosinberg, M. L. & Tarjus, G. Local Mean-Field Study of Capillary Condensation in Silica Aerogels. *Physical Review E* **68,** 061504. doi:10.1103/PhysRevE.68.061504 (17th Dec. 2003).
18. Nattermann, T. in *Spin Glasses and Random Fields Series on Directions in Condensed Matter Physics* Volume 12, 277–298 (WORLD SCIENTIFIC, Dec. 1997). ISBN: 978-981-02-3183-5. doi:10.1142/9789812819437_0009.
19. Fischer, K. H. & Hertz, J. A. in *Spin Glasses* 1–14 (Cambridge University Press, Cambridge, 1991). doi:10.1017/CBO9780511628771.002.
20. Maritan, A., Cieplak, M., Swift, M. R. & Banavar, J. R. Spin-Flip Avalanches and Dynamics of First Order Phase Transitions. *Physical Review Letters* **72,** 946–946. doi:10.1103/PhysRevLett.72.946 (7th Feb. 1994).
21. Pérez-Reche, F. J. & Vives, E. Spanning Avalanches in the Three-Dimensional Gaussian Random-Field Ising Model with Metastable Dynamics: Field Dependence and Geometrical Properties. *Physical Review B* **70,** 214422. doi:10.1103/PhysRevB.70.214422 (20th Dec. 2004).
22. Pérez-Reche, F. J. & Vives, E. Finite-Size Scaling Analysis of the Avalanches in the Three-Dimensional Gaussian Random-Field Ising Model with Metastable Dynamics. *Physical Review B* **67,** 134421. doi:10.1103/PhysRevB.67.134421 (24th Apr. 2003).
23. Detcheverry, F., Kierlik, E., Rosinberg, M. L. & Tarjus, G. Helium Condensation in Aerogel: Avalanches and Disorder-Induced Phase Transition. *Physical Review E* **72,** 051506. doi:10.1103/PhysRevE.72.051506 (29th Nov. 2005).

24. Mandelbrot, B. B. *The Fractal Geometry of Nature* (W. H. Freeman and Comp., New York, 1983).

25. Falconer, K. J. in *Fractal Geometry : Mathematical Foundations and Applications* Third edition., 66–82 (2014). ISBN: 978-1-119-94239-9.

26. Bunde, A. & Havlin, S. in *Fractals in Science* (eds Bunde, A. & Havlin, S.) 1–26 (Springer Berlin Heidelberg, Berlin, Heidelberg, 1994). ISBN: 978-3-642-77953-4. doi:10.1007/978-3-642-77953-4_1.

27. Coniglio, A., De Arcangelis, L. & Herrmann, H. J. Fractals and Multifractals: Applications in Physics. *Physica A: Statistical Mechanics and its Applications* **157,** 21–30. ISSN: 0378-4371. doi:10.1016/0378-4371(89)90272-0 (1st May 1989).

28. Meakin, P. *Fractals, Scaling and Growth Far from Equilibrium* ISBN: 0-521-18981-0 (Cambridge University Press, USA, 1998).

29. Baker, G. L. & Gollub, J. P. *Chaotic Dynamics: An Introduction* 2nd ed. ISBN: 978-0-521-47685-0. doi:10.1017/CBO9781139170864 (Cambridge University Press, Cambridge, 1996).

30. Ito, N. & Suzuki, M. Fractal Configurations of the Two- and Three-Dimensional Ising Models at the Critical Point. *Progress of Theoretical Physics* **77,** 1391–1401. ISSN: 0033-068X. doi:10.1143/PTP.77.1391 (1st June 1987).

31. Edgar, G. A. in *Measure, Topology, and Fractal Geometry / Gerald Edgar.* 2nd ed., 165–224 (Springer, New York, 2008). ISBN: 978-0-387-74748-4.

32. Falconer, K. J. in *Fractal Geometry : Mathematical Foundations and Applications* Third edition., 44–65 (2014). ISBN: 978-1-119-94239-9.

33. Falconer, K. J. in *Fractal Geometry : Mathematical Foundations and Applications* Third edition., xix–xxx (2014). ISBN: 978-1-119-94239-9.

34. Meakin, P. Diffusion-Limited Aggregation in Three Dimensions: Results from a New Cluster-Cluster Aggregation Model. *Journal of Colloid and Interface Science* **102,** 491–504. ISSN: 0021-9797. doi:10.1016/0021-9797(84)90252-2 (1st Dec. 1984).

35. Herrmann, H. J. Geometrical Cluster Growth Models and Kinetic Gelation. *Physics Reports* **136,** 153–224. ISSN: 0370-1573. doi:10.1016/0370-1573(86)90047-5 (1st Apr. 1986).

36. Niblett, S. *A Computational Study of Capillary Condensation in Disordered Porous Media* (University of Cambridge, 2014).

37. Cepitis, R. *Avalanche Statistics in Vycor and Aerogel* (University of Cambridge, 2020).

38. Barahona, F. Finding Ground States in Random-Field Ising Ferromagnets. *Journal of Physics A: Mathematical and General* **18,** L673–L675. ISSN: 0305-4470. doi:10.1088/0305-4470/18/11/008 (Aug. 1985).

39. Alava, M., Duxbury, P., Moukarzel, C. & Rieger, H. in *Phase Transitions and Critical Phenomena* 143–317 (Elsevier, 2001). ISBN: 978-0-12-220318-3. doi:10.1016/S1062-7901(01)80009-4.

40. Ford, L. R. & Fulkerson, D. R. *Flows in Networks* ISBN: 978-1-4008-7518-4. doi:10.1515/9781400875184 (Princeton University Press, 1962).

41. Boykov, Y. & Kolmogorov, V. An Experimental Comparison of Min-Cut/Max- Flow Algorithms for Energy Minimization in Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **26,** 1124–1137. ISSN: 1939-3539. doi:10.1109/TPAMI.2004.60 (Sept. 2004).

42. Crow, F. C. *Summed-Area Tables for Texture Mapping* in *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques* (Association for Computing Machinery, New York, NY, USA, 1st Jan. 1984), 207–212. ISBN: 978-0-89791-138-2. doi:10.1145/800031.808600.

43. Viola, P. & Jones, M. J. Robust Real-Time Face Detection. *International Journal of Computer Vision* **57,** 137–154. ISSN: 1573-1405. doi:10.1023/B:VISI.0000013087.49260.fb (1st May 2004).

44. Tapia, E. A Note on the Computation of High-Dimensional Integral Images. *Pattern Recognition Letters* **32,** 197–201. ISSN: 0167-8655. doi:10.1016/j.patrec.2010.10.007 (15th Jan. 2011).

45. Clauset, A., Shalizi, C. R. & Newman, M. E. J. Power-Law Distributions in Empirical Data. *SIAM Review* **51,** 661–703. ISSN: 0036-1445. doi:10.1137/070710111. arXiv: 0706.1062 (4th Nov. 2009).

46. Stumpf, M. P. H. & Porter, M. A. Critical Truths About Power Laws. *Science* **335,** 665–666. doi:10.1126/science.1216142 (10th Feb. 2012).

47. Stanley, H. E. Scaling, Universality, and Renormalization: Three Pillars of Modern Critical Phenomena. *Reviews of Modern Physics* **71,** S358. ISSN: 00346861. doi:10.1103/RevModPhys.71.S358 (1st Mar. 1999).

48. Fisher, M. E. & Barber, M. N. Scaling Theory for Finite-Size Effects in the Critical Region. *Physical Review Letters* **28,** 1516–1519. doi:10.1103/PhysRevLett.28.1516 (5th June 1972).

49. Cardy, J. in *Current Physics–Sources and Comments* 1–7 (Elsevier, 1988). ISBN: 978-0-444-87109-1. doi:10.1016/B978-0-444-87109-1.50006-6.

50. Kawashima, N. & Ito, N. Critical Behavior of the Three-Dimensional ±J Model in a Magnetic Field. *Journal of the Physical Society of Japan* **62,** 435–438. ISSN: 13474073. doi:10.1143/JPSJ.62.435 (Feb. 1993).

51. Bhattacharjee, S. M. & Seno, F. A Measure of Data Collapse for Scaling. *Journal of Physics A: Mathematical and General* **34,** 6375. ISSN: 0305-4470. doi:10.1088/0305-4470/34/33/302. arXiv: cond-mat/0102515 (10th Aug. 2001).

52. Houdayer, J. & Hartmann, A. K. Low-Temperature Behavior of Two-Dimensional Gaussian Ising Spin Glasses. *Physical Review B - Condensed Matter and Materials Physics* **70,** 014418. ISSN: 01631829. doi:10.1103/PhysRevB.70.014418. arXiv: cond-mat/0402036 (16th July 2004).

53. Deming, W. E. in *Statistical Adjustment of Data* 1st ed. (John Wiley & Sons ; Chapman & Hall, New York : London, 1943).

B Code

B.1 Wrapper program

```
!  Filename: /home/jake/Documents/Project/avalanche_fractal_analysis/main.f95
!  Path: /home/jake/Documents/Project/avalanche_fractal_analysis
!  Created Date: Monday, November 15th 2021, 12:11:58 am
!  Author: Jake Skelton

!  Copyright (c) 2021 Jake Skelton

! IO wrapper for sandbox analysis. Takes list of fluid coordinates involved in ! avalanche, and matrix
! 'occupation arrays' to sandbox.f95, which does the legwork. Then, this program
! also handles outputting results to the outfile.
! Calling procedure:
!   main <f_file> <mat_file> <lmax> <outfile> [options]
! Dictionary:
!   <f_file>:   Input file path, containing fractal coordinates
!   <mat_file>: File path for matrix coordinates (complementary representation)
!   <lmax>:     System size
!   <outfile>:  Output file path, will get average masses and dimension written
!                   to it. If output not desired, use dev/null
!   [options]:  Remaining flags:
!                   [sample size] - Any numeric value following <outfile> will
!                       be interpreted as the percentage of sites in the
!                       avalanche to use in the random sample of origins.
!                       Must include a decimal point.
!                       Passing '100.0' is equivalent to leaving absent
!                   v - verbose mode

module input_output
    implicit none

contains

integer*8 function get_file_length(file)
    ! Simple function to run through all the lines of a file and count them

    character(256), intent(in) :: file
    integer, parameter :: fu = 100
    integer :: filestatus = 0
    integer*8 :: lines = 0

    open(unit=fu, file=file, status='old', action='read')

    do
        read(fu, *, iostat=filestatus)
        if (filestatus /= 0) exit   ! Stop at EOF
        lines = lines + 1
    end do
    close(fu)

    get_file_length = lines

end function get_file_length

subroutine read_coords(file, length, D, arr, coords)

    character(256), intent(in) :: file
    integer*8, intent(in) :: length
    integer, intent(in) :: D
    integer, intent(out) :: arr(D, D, D)
    integer, intent(out) :: coords(length, 3)

    character(256) :: err
    integer, parameter :: fu = 200  ! File unit variable
    real*8 :: x, y, z
    integer*8 :: i
    integer :: intco(3), flag, filestatus
```

```
    open(unit=fu, file=file, status='old', action='read', iostat=flag, &
        iomsg=err)

    if (flag /= 0) write(*,*) err

    arr = 0
    do i = 1, length
        read(fu, *, iostat=filestatus) x, y, z
        if (filestatus /= 0) exit   ! Stop at EOF

        ! Simple cubic --> BCC & start from 1
        intco = floor([x, y, z]) + 1
        arr(intco(1), intco(2), intco(3)) = &
            arr(intco(1), intco(2), intco(3)) + 1
        coords(i,:) = intco
    end do
    close(fu)

end subroutine read_coords

character(256) function filefrompath(filepath)
    ! From path "parent_directory/file.ext" extract just "file.ext"
    ! NB Unix slash

    character(256) :: filepath
    integer :: marker

    marker = index(filepath, '/', back=.true.)
    filefrompath = trim(filepath(marker+1:))

end function filefrompath

integer*8 function codefromfile(filename)
    ! Slurp all the numeric characters out of an input string, and turn them
    ! into a big integer, preserving their order

    character(256), intent(in) :: filename

    character(256) :: numstring
    integer :: i, j

    j = 0
    do i = 1, len_trim(filename)
        if (lle('0', filename(i:i)) .and. lle(filename(i:i), '9')) then
            j = j + 1
            numstring(j:j) = filename(i:i)
        end if
    end do
    read(numstring(:j), '(i8)') codefromfile

end function codefromfile

subroutine random_sample(arr, seed, outarr)

    integer, intent(inout) :: arr(:,:)
    integer*8, intent(in) :: seed
    integer, intent(out) :: outarr(:,:)

    integer :: seed_size
    integer*8 :: asize, oasize, i, j
    integer, allocatable :: temp(:), seed_array(:)
    real*8, allocatable :: trials(:)

    asize = size(arr, 1)
    oasize = size(outarr, 1)
    allocate(temp(size(arr, 2)), trials(asize))

    ! Make random numbers
```

```
    call random_seed(size=seed_size)
    allocate(seed_array(seed_size))
    seed_array = seed
    call random_seed(put=seed_array)
    call random_number(trials)

    ! Trim arr down to outarr by doing Fisher-Yates shuffle and selecting first
    ! oasize elements
    do i = 1, asize - 1
        j = i + floor((asize - i) * trials(i))
        ! Axchange arr(i, :) and arr(j,:)
        temp = arr(j, :)
        arr(j, :) = arr(i, :)
        arr(i, :) = temp
    end do
    outarr = arr(:oasize, :)

end subroutine random_sample

subroutine writetofile(outfile, infile, &
    df, errdf, numsites, mp, vp, mf, vf, verbose)
    ! Ouput results of sandbox analysis to file. Format is as follows
    ! [infile_name] [df] [errdf] [aval_size] [mp_1] [vp_1] [mf_1] [vf_1] ... [mp_lmax] [vp_lmax] [mf_lm
    ! i.e. the columns following column 4 come in groups of four, each
    ! containing (mp, vp, mf, vf) for one of the lmax bounding boxes

    character(256), intent(in) :: infile
    character(256), intent(in) :: outfile
    real*8, intent(in) :: df, errdf
    integer*8, intent(in) :: numsites
    real*8, intent(in) :: mp(:), vp(:), mf(:), vf(:)
    logical, intent(in), optional :: verbose

    character(256) :: fmt, vfmt, hdrfmt
    integer, parameter :: fu = 1000
    integer :: l, lmax

    ! Output width depends on number of data points
    lmax = size(mp)
    write(fmt, '("(a48, 2f8.4, i10, ", i8, "es18.8)")') 4*lmax

    open(unit=fu, file=outfile, action='write', position='append')
    write(fu, fmt) infile, df, errdf, numsites, &
        (mp(l), vp(l), mf(l), vf(l), l = 1, lmax)
    close(fu)

    if (present(verbose) .and. verbose) then
        vfmt = '(i6, 4es18.8)'
        hdrfmt = '(a6, 4a18)'
        write(*, hdrfmt) 'l', 'mp', 'vp', 'mf', 'vf'
        do l = 1, lmax
            write(*, vfmt) l, mp(l), vp(l), mf(l), vf(l)
        end do
    end if

end subroutine writetofile

end module input_output

program main
! The main coordinator for avalanche fractal analysis. Reads input file
! given as command line argument, converts coordinates to occupancy, sends
! for sandbox algorithm analysis, receives statistics back, writes to
! output file.
use input_output
use sandbox
use lin_reg
use iso_fortran_env, only: error_unit
implicit none
```

```
integer :: i, lmax, numargs
integer*8 :: j, numfluid, nummat, ransize
integer, allocatable :: A(:,:,:), T(:,:,:), &
    flucoords(:,:), matcoords(:,:), samplecoords(:,:)
real*8 :: intcpt, df, errintcpt, errdf, ranprop = 1.0
real*8, allocatable :: mp(:), vp(:), mf(:), vf(:)
character(256) :: infile, matfile, outfile, filename, temp
logical :: verbose = .false.

! Interpret command line arguments
numargs = command_argument_count()
call get_command_argument(1, infile)
call get_command_argument(2, matfile)
call get_command_argument(3, temp)
read(temp, '(i8)') lmax
call get_command_argument(4, outfile)
if (numargs > 4) then
    do i = 5, numargs
        call get_command_argument(i, temp)
        ! Only check arguments if not found yet
        if (.not. verbose) verbose = (index(temp, 'v') /= 0)
        if (verify(temp, ' .0123456789') == 0) then
            ! User has passed size of random sample
            read(temp, '(f8.4)') ranprop
            ranprop = ranprop / 100.0
        else
            ranprop = 1.0
        end if
    end do
    if (verbose) write(*,*) 'Entering verbose mode'
end if

! Read file lengths and allocate arrays
filename = filefrompath(infile)
numfluid = get_file_length(infile)  ! Total number of fluid coords
nummat = get_file_length(matfile)   ! "             " matrix "  "
ransize = floor(ranprop * numfluid)
allocate(&
    flucoords(numfluid, 3),&    ! Fluid coordinates list
    matcoords(nummat, 3), &     ! Matrix coordinates list
    A(lmax, lmax, lmax), &      ! Fluid occupation array
    T(lmax, lmax, lmax), &      ! Matrix occupation array
    mp(lmax), &                 ! Mean pore space in bboxes of size l
    vp(lmax), &                 ! Variance in pore cell number in l-bboxes
    mf(lmax), &                 ! Mean fluid cells in bboxes of size l
    vf(lmax), &                 ! Variance in fluid cell number in l-bboxes
    samplecoords(ransize, 3), & ! Random sample of fluid coords
)

! Read in fluid and matrix coordinates, filling coord and occ arrays
call read_coords(infile, numfluid, lmax, A, flucoords)
call read_coords(matfile, nummat, lmax, T, matcoords)

! Reduce flucoords to a random subset of size ranprop * numfluid
if (ransize < numfluid) then
    call random_sample(flucoords, codefromfile(filename), samplecoords)
else
    samplecoords = flucoords
end if

! Run core code from sandbox.f95
call main_loop(samplecoords, A, T, mp, vp, mf, vf)

! Quick and dirty least squares straight-line fit
call lst_sq_fit(log10(mp), log10(mf), intcpt, df, errintcpt, errdf)

if (verbose) then
    do i = 1, lmax
```

```
        write(*,'("\r\n z = ", i4)') i
        do j = 1, lmax
            write(*, '(100i2)') A(:, j, i)
        end do
    end do
end if

! Output results to outfile
call writetofile(outfile, filename, df, errdf, ransize, mp, vp, mf, vf, verbose)

! Give a little helpful output to stdout
write(*,'(a48, i8, f14.8, f14.8)') &
        filename, ransize, df, errdf

end program main
```

B.2 Module for sandbox analysis numerical routines

```
!   Filename: sandbox.f95
!   Path: /home/jake/Documents/Project/avalanche_fractal_analysis
!   Created Date: Wednesday, October 27th 2021, 11:09:55 pm
!   Author: Jake Skelton

!   Copyright (c) 2021 Jake Skelton

! Core code to perform sandbox analysis on a single avalanche. To be called by
! program main stored in main.f95

module sandbox

implicit none
contains

subroutine main_loop(coords, A, T, mp, vp, mf, vf)

   integer, intent(in) :: &
      coords(:,:), & ! BCC coordinates of avalanche fluid cells
      A(:,:,:), &    ! Array representation of avalanche
      T(:,:,:)       ! Array representation of pore space
   real*8, intent(out) :: &
      mp(:), &       ! Mean number of pore cells in bounding box of linear size
                     !  [index]
      vp(:), &       ! As above, but variance
      mf(:), &       ! Mean number of fluid cells in bounding box of linear
                     !  size [index]
      vf(:)          ! As above, but variance

   integer, allocatable :: tessA(:,:,:), tessT(:,:,:), ii(:,:,:), iiT(:,:,:)
   integer :: x, y, z, coord(3), lmax, l
   integer*8 :: n, numsites, f, p

   ! Get important sizes from arguments
   numsites = size(coords, 1)
   lmax = size(A, 1)
   ! [1, lmax]^3 references the centre of the tesselated system
   allocate(tessA(-lmax+1  : 2*lmax, -lmax+1 : 2*lmax, -lmax+1 : 2*lmax), &
            tessT(-lmax+1  : 2*lmax, -lmax+1 : 2*lmax, -lmax+1 : 2*lmax), &
               ii(-lmax    : 2*lmax, -lmax   : 2*lmax, -lmax   : 2*lmax), &
             iiT(-lmax     : 2*lmax, -lmax   : 2*lmax, -lmax   : 2*lmax))
              ! One greater on each dimension to prevent OOB

   ! Tesselate A 3x3x3 times to deal with periodic BCs. Then, make
   ! integral image from it
   do x = -lmax + 1, lmax + 1, lmax
      do y = -lmax + 1, lmax + 1, lmax
         do z = -lmax + 1, lmax + 1, lmax
            tessA(x : x+lmax-1, y : y+lmax-1, z : z+lmax-1) = A
            tessT(x : x+lmax-1, y : y+lmax-1, z : z+lmax-1) = T
         end do
```

```
      end do
   end do
   call makeii(lmax, tessA, ii)
   call makeii(lmax, tessT, iiT)

   ! Calculate total mass in system
   mp = 0.; vp = 0.; mf = 0.; vf = 0.
# ifdef DEBUG
   write(*, "(a8, i8, 5a4)") 'origin /', numsites, 'x', 'y', 'z', 'l'
# endif
   ! Main loop
   do n = 1, numsites
      coord = coords(n, :)

      ! Can save effort for single-element boxes
      f = A(coord(1), coord(2), coord(3))
      p = T(coord(1), coord(2), coord(3))
      call iterate_stats(n, mf(1), vf(1), f)
      call iterate_stats(n, mp(1), vp(1), p)

      do l = 2, lmax
#ifdef DEBUG
            write(*, '(a, i16, 5i4)', advance='no') achar(13), n, coord, l
#endif
         call iisum(lmax, l, ii, coord, f)
         call iisum(lmax, l, iiT, coord, p)
         call iterate_stats(n, mf(l), vf(l), f)
         call iterate_stats(n, mp(l), vp(l), p)

      end do
   end do

   ! E(x^2) - (E(x))^2
   vp = vp - mp**2
   vf = vf - mf**2

# ifdef DEBUG
   write(*,*)
# endif

end subroutine main_loop

subroutine makeii(D, A, ii)
   ! Construct the integral image (Viola, Jones 2004) from a given array.

   ! Had to make the argument list a bit ugly to retain funky subscripts
   integer, intent(in) :: D      ! 1/3 * sidelength of A
      ! Array ('image') we wish to compute sums on
   integer, intent(in), dimension(-D+1:2*D, -D+1:2*D, -D+1:2*D) :: A
      ! Output object for integral image
   integer, intent(out),dimension(-D:2*D, -D:2*D, -D:2*D) :: ii

   integer :: x, y, z

   ii = 0
   do x = -D + 1, 2*D      ! Recall ii has non-standard subscripts
      do y = -D + 1, 2*D
         do z = -D + 1, 2*D
            ii(x, y, z) = A(x, y, z) &
               + ii(x-1, y, z)   + ii(x, y-1, z)   + ii(x, y, z-1) &
               - ii(x-1, y-1, z) - ii(x-1, y, z-1) - ii(x, y-1, z-1) &
               + ii(x-1, y-1, z-1)
         end do
      end do
   end do

end subroutine makeii

subroutine iisum(lmax, l, ii, centre, mass)
```

```
   ! Compute a box sum on an array by utilising the integral image
   ! representation. This method performs independently of box size.

   integer, intent(in) :: lmax
      ! The integral image array
   integer, dimension(-lmax:2*lmax,-lmax:2*lmax,-lmax:2*lmax), intent(in) :: ii
   integer, intent(in) :: l            ! Sidelength of box to sum
   integer, intent(in) :: centre(3)    ! Centre of box
   integer*8, intent(out) :: mass      ! Where the total will be put

   integer :: lb(3)

   if (mod(l, 2) == 0) then       ! Test even
      lb = centre - (l/2 - 1) - 1   ! Asymmetric bounds
   else
      lb = centre - (l - 1)/2 - 1    ! Symmetric bounds
   end if

   ! Can be expressed more succintly but this needs to go fast
   mass =  ii(lb(1) + l, lb(2) + l, lb(3) + l) &
         - ii(lb(1) + l, lb(2) + l, lb(3)    ) &
         - ii(lb(1) + l, lb(2)    , lb(3) + l) &
         - ii(lb(1)    , lb(2) + l, lb(3) + l) &
         + ii(lb(1) + l, lb(2)    , lb(3)    ) &
         + ii(lb(1)    , lb(2) + l, lb(3)    ) &
         + ii(lb(1)    , lb(2)    , lb(3) + l) &
         - ii(lb(1)    , lb(2)    , lb(3)    )

end subroutine iisum

subroutine iterate_stats(n, mean, sumsq, datum)
! Update mean and sum of squares using 'datum' when moving from n - 1 to n
! data points. i.e. this should first be called with n=1.

   integer*8, intent(in) :: n
   integer*8, intent(in) :: datum
   real*8, intent(inout) :: mean, sumsq

   mean = (datum + (n - 1)*mean) / n
   sumsq = (datum**2 + (n - 1)*sumsq) / n

end subroutine iterate_stats

end module sandbox
```