Percolation thresholds for discrete-continuous models with nonuniform probabilities of bond formation

Bartłomiej Szczygieł, Marek Dudyński, Kamil Kwiatkowski, Maciej Lewenstein, Gerald John Lapeyre, Jr., and Jan Wehr

Phys. Rev. E **93**, 022127 — Published 18 February 2016

DOI: 10.1103/PhysRevE.93.022127

Percolation thresholds for discrete–continuous models with non-uniform probabilities of bond formation

Bartłomiej Szczygieł[*]

College of Inter-Faculty Individual Studies in Mathematics and Natural Sciences, University of Warsaw, Żwirki i Wigury 93, 02-089 Warsaw, Poland

Marek Dudyński[†]

Modern Technologies and Filtration, Przybyszewskiego 73/77 lok. 8, 01-824 Warsaw, Poland

Kamil Kwiatkowski[‡]

Institute of Theoretical Physics, Faculty of Physics, University of Warsaw, Pasteura 5, 02-093 Warsaw, Poland and Interdisciplinary Centre for Mathematical and Computational Modeling, University of Warsaw, Prosta 69, 00-838 Warsaw, Poland

Maciej Lewenstein[§]

ICFO-Institut de Ciències Fotòniques, The Barcelona Institute of Science and Technology, Av. Carl Friedrich Gauss 3, 08860 Barcelona, Spain and ICREA-Institució Catalana de Recerca i Estudis Avançats, Lluis Campanys 23, 08010 Barcelona, Spain

Gerald John Lapeyre Jr[¶]

Spanish National Research Council (IDAEA-CSIC), E-08034 Barcelona, Spain and ICFO-Institut de Ciències Fotòniques, The Barcelona Institute of Science and Technology, Av. Carl Friedrich Gauss 3, 08860 Barcelona, Spain

Jan Wehr[**]

Department of Mathematics, University of Arizona, Tucson AZ 85721, USA

(Dated: January 25, 2016)

Abstract

We introduce a class of discrete-continuous percolation models and an efficient Monte Carlo algorithm for computing their properties. The class is general enough to include well-known discrete and continuous models as special cases. We focus on a particular example of such a model, a nanotube model of disintegration of activated carbon. We calculate its exact critical threshold in 2d and obtain a Monte Carlo estimate in 3d. Furthermore, we use this example to analyze and characterize the efficiency of our algorithm, by computing critical exponents and properties, finding that it compares favorably to well-known algorithms for simpler systems.

* bartlomiej.szczygiel@students.mimuw.edu.pl

† marek.dudynski@mtf.pl

‡ kamil.kwiatkowski@fuw.edu.pl

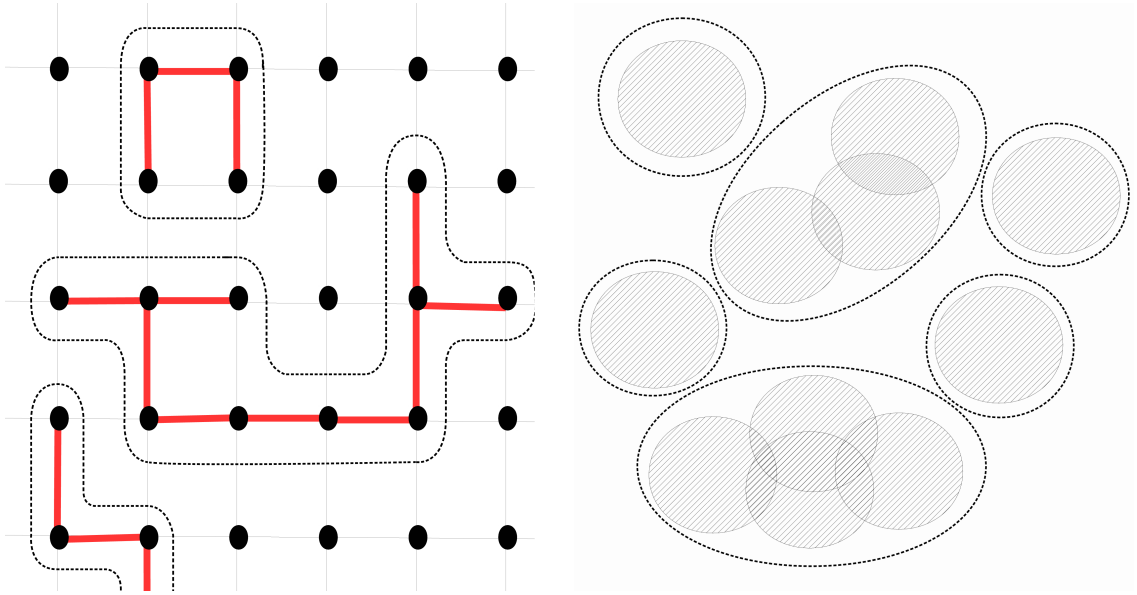
§ maciej.lewenstein@icfo.es

¶ john.lapeyre@icfo.es

** wehr@math.arizona.edu

I. INTRODUCTION

Two basic types of percolation models are discrete and continuous percolation [1, 2]. In the discrete case, a lattice is given and its bonds (edges) are open, or its sites (vertices) are occupied, with a probability p, which is the relevant parameter of the model. Depending on the case, we speak of *bond percolation* or *site percolation*. The local random variables, which determine bond openness or site occupations, define global connections and the main focus of the theory is the phenomenon of *percolation*, i.e. the appearance of an infinite cluster (or, in some models: of infinite clusters) of connected bonds or sites. In continuum models the positions of percolating objects themselves are chosen at random in space and the connections are determined solely by the realization of the objects [3]. A parameter η playing a role analogous to p is usually defined as the expected value of the local density of the objects. We refer to η or p as the model parameter, and use the symbol ρ for the generalized discrete-continuous case. In the discrete approach, one can also generate the lattice randomly, and then open all its edges with the same probability, which does not depend on the random geometry. Classical examples of discrete and continuum percolation are presented in Fig. 1.



(a) (b)

FIG. 1. (Color online) Examples of discrete (a) and continuum (b) percolation: (a) bond percolation on the square lattice, b) discs in the plane. Clusters are delineated in both cases.

However, there are instances when complexities of percolation phenomena are beyond the scope of these two basic types of percolation model. A simple example is a system of roads, in which the width of a road is described by the weight of the corresponding edge and the traffic intensity corresponds to the percolation parameter. In this situation the probability of a road connection between two points being open is a function of both these parameters [4, 5]. Another interesting case, the so-called radio tower model [6], is obtained by modifying the disc percolation model [7]. In this model we first randomly distribute in the plane points (towers) which become the centers of discs with fixed radius R. No pair of towers can communicate if they are separated by a distance greater than R, which is the parameter of the model. We set the probability that a connection (an open bond) exists between a pair of towers as $p_{\text{bond}} = \max(0, 1 - d/R)$, where d is a distance between the two points. We look for the critical value of R at which an infinite cluster appears.

These models have two things in common: their geometry is random and the possible connections in the system are determined by a random variable, whose distribution is defined by both the geometry and the discrete-continuous model parameter ρ. Models in this discrete-continuous class are thus described by a random graph with weighted edges, where the probability of a connection depends both on the model parameter, and on the weight of edges, dictated by the geometry of the graph realization. The parameter of discrete-continuous percolation models ρ is more general than in the discrete case. It is not necessarily in the range $[0, 1]$, or even bounded. It includes R from the radio tower model and the probability p of a bond connection in discrete percolation models, as well as other model-specific parameters.

The present paper consists of three contributions. First, we introduce a general framework to describe discrete-continuous percolation. This framework is general enough to include both standard discrete and continuous models as special cases, as well as hybrid models such as those mentioned above. Secondly, we use this framework to study tube-based models of activated carbon that have received attention recently. We calculate the critical threshold exactly in two dimensions and via Monte Carlo in three dimensions. Finally, we introduce a highly efficient Monte Carlo (MC) algorithm inspired by the Newman-Ziff algorithm (NZ) for estimating all the relevant properties of general discrete-continuous percolation models. This algorithm shares with NZ a union-find algorithm that is practically optimal for cluster-building [8, 9]. In the discrete-continuous case different bonds are open with differ-

ent probabilities. Since in such case the use of the 'micro-canonical' approach proposed in Ref. [9] is difficult at best, we introduce an alternative method that also efficiently simulates many values of model parameters in a single run.

The particular example of discrete-continuous percolation that we focus on in this paper consists of parallel random tubes connected randomly by bonds. These models, merging the characteristics of discrete and continuous percolation, are motivated by the structures and properties of activated carbon [10, 11]. When charcoal is transformed into activated carbon, the initial structure (skeleton) of wood, composed of parallel cylinders, persists, but their walls are transformed into more carbonic compounds. Simultaneously, the microscopic structure of the walls becomes much more complex. Fine micro-porous substructures [12] are formed, and lead to a rapid increase of internal surface (specific surface area). If the process is not stopped, the structure of the material finally breaks down, making it collapse into fine dust, which burns into a small amount of ash [13, 14]. Several models were developed to explain the complicated micro-structures observed in charcoal and activated carbon [15]. Most proposed models are based on carbon nanotubes [10, 11, 16, 17], but various other forms of carbon potentially building such micro-structures were also considered, including graphene ribbons (Jenkins-Kawamura model), fullerenes (Harris model [18, 19]), stacked graphite [20] or graphene [15], carbon onions [21]. More concretely, we assume that the skeleton walls are made of a collection of parallel tubes representing nanopipes of varying lengths. These nanopipes form an inhomogeneous lattice bound together by amorphous carbon connections. We assume that during gasification the amorphous carbon is reacting and the bonds are removed. The bond removal leads to disconnecting more and more nanopipes, leading to disintegration into small clusters and, finally, to breakdown of the percolating skeleton.

In order to compute properties of discrete-continuous types of percolation models such as tube-based model, efficient algorithms have to be developed. To be fully useful, an algorithm should not only efficiently compute the percolation threshold for inhomogeneous lattices but also gather other statistics of the process, such as calculating critical parameters and cluster density distributions. In this paper we present and analyze such an algorithm. We applied the algorithm to the family of tube-based models, computing their critical parameters and cluster density distributions in two and three dimensions. We have positively verified the algorithm by comparison with the exact solutions of the percolation threshold in two di-

mensional case. We analyzed its convergence properties, as well as memory and CPU use, showing them to be similar to the computational costs of NZ.

Of course other lattice-based algorithms may potentially be applied to discrete-continuous percolation including the well-known Hoshen-Kopelman algorithm or its extended versions [22–24] and the Leath-Alexandrowicz [25, 26] algorithms. Modern and efficient versions of these algorithms could indeed be developed for discrete-continuous models. However, an important requirement for the present study is efficient simulation of several values of ρ in a single run. The Hoshen-Kopelman algorithm, for instance, is better suited for simulating extremely large lattices for a single value of ρ.

It has to be stressed that our percolation algorithm is not dependent on the details of the particular tube model of activated carbon studied here. It can be applied with the same efficiency to a broad class of inhomogeneous lattices and discrete-continuous percolation systems that fall within our general framework, such as the radio tower and road network models mentioned above.

The remainder of the paper is structured as follows. In Sec. II we describe the tube-based percolation model. In Sec. III we then describe our Monte Carlo algorithm for discrete-continuous percolation, which allows us to treat the inhomogeneity of the lattice inherent in our model. In Sec. IV, we validate our algorithm. We compute the exact critical threshold of the tube model in two dimensions and verify that our algorithm reproduces this value to high accuracy. We further test the algorithm by computing critical exponents that verify that the tube model lies in the standard percolation universality class. Finally, we use the algorithm to obtain new results for the three dimensional problem in Sec. V.

II. PERCOLATION MODEL

We now describe the particular example of discrete-continuous percolation that we will study in this paper: the tube-based model. To define the model precisely in two-dimensions, we proceed in three steps:

- we start from n parallel (vertical, for definiteness) lines of length L.
- we use n independent Poisson processes with the same parameter μ_1 to divide the lines into segments, called tubes. That is, the points of the Poisson process are the

endpoints of the segments.

- we introduce bonds between each pair of adjacent lines and in this manner the connections between tubes are established. The bonds are generated by independent Poisson processes with parameter μ_2. In this case the points of the Poisson process determine the positions of the horizontal bonds.

A sample realization of the tube model is presented in Fig. 2. The three dimensional model

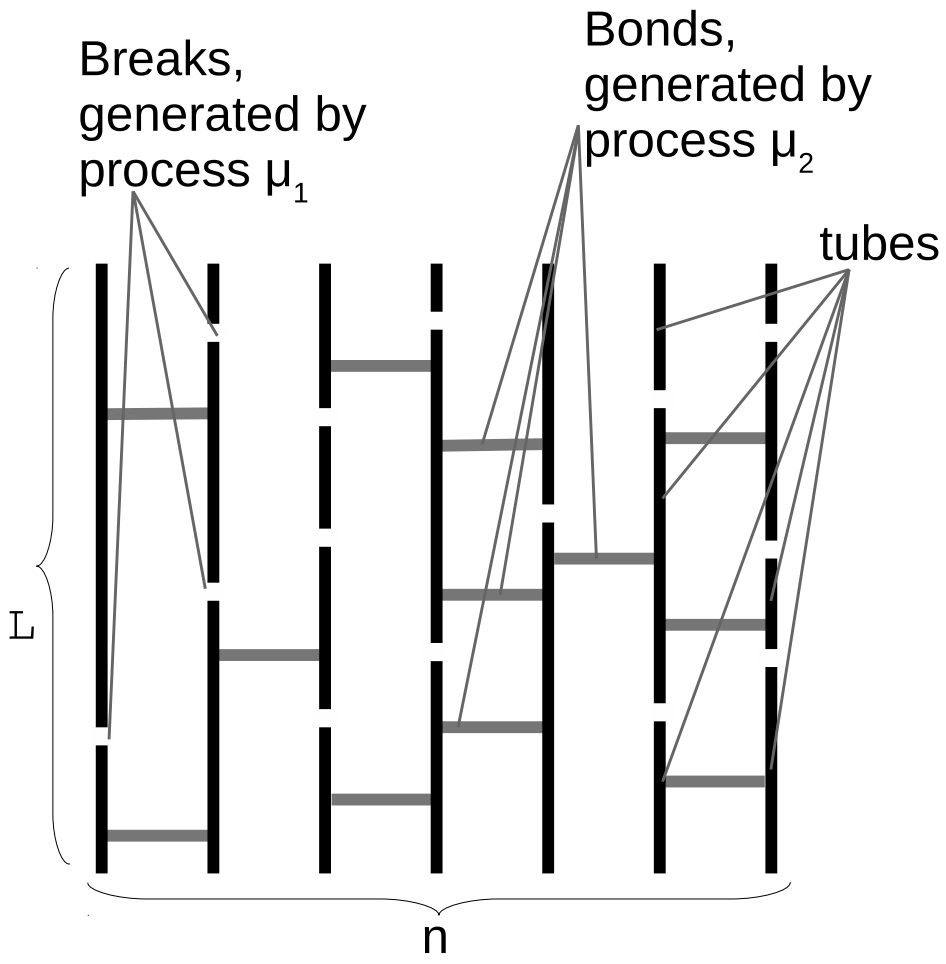


FIG. 2. (Color online) An example of a realization of the two dimensional graph described by a set of four parameters $\{L, n, \mu_1, \mu_2\}$.

is defined similarly. First, we introduce a set of lines of length L passing through the points of a square lattice and perpendicular to plane of this lattice. Then we follow the procedure for 2D case, dividing lines into segments and generating bonds between each pair of adjacent

lines. "Adjacent" is defined here using the nearest-neighbor connections on the underlying square lattice, so that in the 2D case a line not lying on the boundary has two adjacent lines while in the 3D case it has four. All Poisson processes in the above definition are assumed independent of each other (in 2, as well as in 3 dimensions). Thus, the tube model consists of parallel tubes of random length connected randomly by bonds whose distribution is defined by the spatial location of the tubes and by the model parameter.

The model is described by four parameters $\{L, n, \mu_1, \mu_2\}$, defining the size of the model (L and n), the tubes described by Poisson processes with the parameter μ_1 and with bonds between these tubes generated by independent Poisson processes with the parameter μ_2. Under rescaling of space in the direction of the lines, the resulting graph is equivalent to the system with parameters $\{L\mu_1, n, 1, \mu_2/\mu_1\}$. We can thus put $\mu_1 := 1$ and $\mu_2 = \mu$, so in the limit when L and n go to infinity at the same rate the model has only one parameter μ. It is worth noting that, while μ is a model parameter for tube-based model, it is not a probability. Such parameters are denoted by ρ in the general discussion, see Sec. III A. For simplicity, in most of the simulations we put $L = n$.

By definition, different segments (tubes) of the same line are not connected to each other. Only tubes lying on adjacent lines may be connected, if one or more open bonds between them are established. A single open bond is sufficient to connect two tubes. This allows one to calculate a connection probability p_{bond_i} between two adjacent tubes in terms of their relative position as follows. Two tubes lying on adjacent lines may only be connected if there is a nonzero overlap h_i between their vertical positions as shown in Fig. 3. The probability that two such tubes are connected by k open bonds is given by the Poisson distribution with parameter μh_i. That is,

$$P(k) = \frac{e^{-\mu h_i}(\mu h_i)^k}{k!}. \tag{1}$$

Tubes are disconnected ($k = 0$) with probability $P(0) = e^{-\mu h_i}$ and thus they are connected with probability

$$p_{\mathrm{bond,i}} = 1 - e^{-\mu h_i}. \tag{2}$$

We emphasize that the overlap h_i depends on the pair of tubes. A sample realization of the two dimensional model is presented in Fig. 4. Groups of connected tubes form clusters marked in Fig. 4 by a single color.

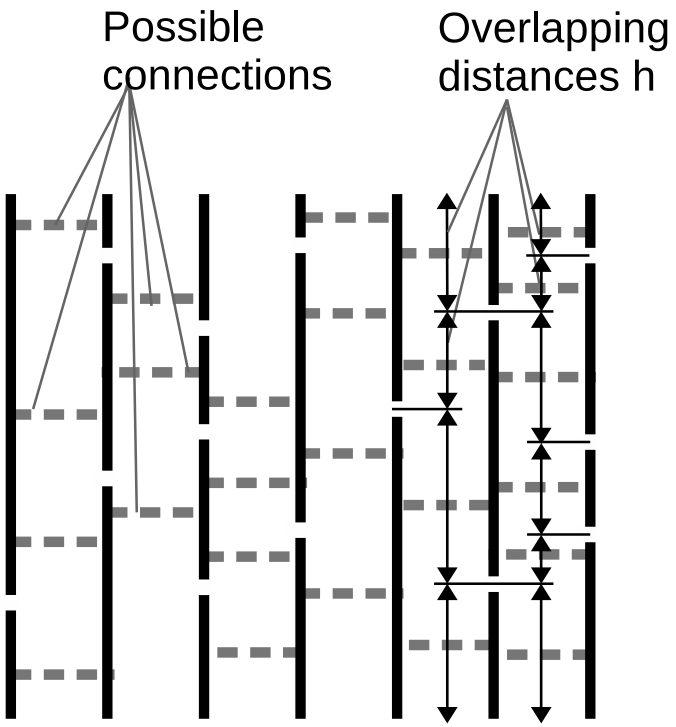


FIG. 3. (Color online) Redefined graph. The overlap between two adjacent tunes is shown by intervals between the arrows.

III. THE ALGORITHM

For the convenience of the reader we summarize below the notation used respectively for general discrete-continuous percolation, for discrete percolation and for the specific tube-based model:

- ρ, p, μ - model parameter;
- $\rho_{\rm c}$, p_c, $\mu_{\rm c}$ - critical point for infinite lattice, the value of the model parameter above which, on the infinite lattice, there is an infinite cluster;
- $\rho_{\rm con,k}$, $p_{\rm con,k}$, $\mu_{\rm con,k}$ - the value of the model parameter when, upon the k-th sub iteration of the algorithm, a connection between lattice edges is first made;
- $\rho_{\rm av}$, $p_{\rm av}$, $\mu_{\rm av}$ - critical point for finite lattice, the expected value of the model parameter

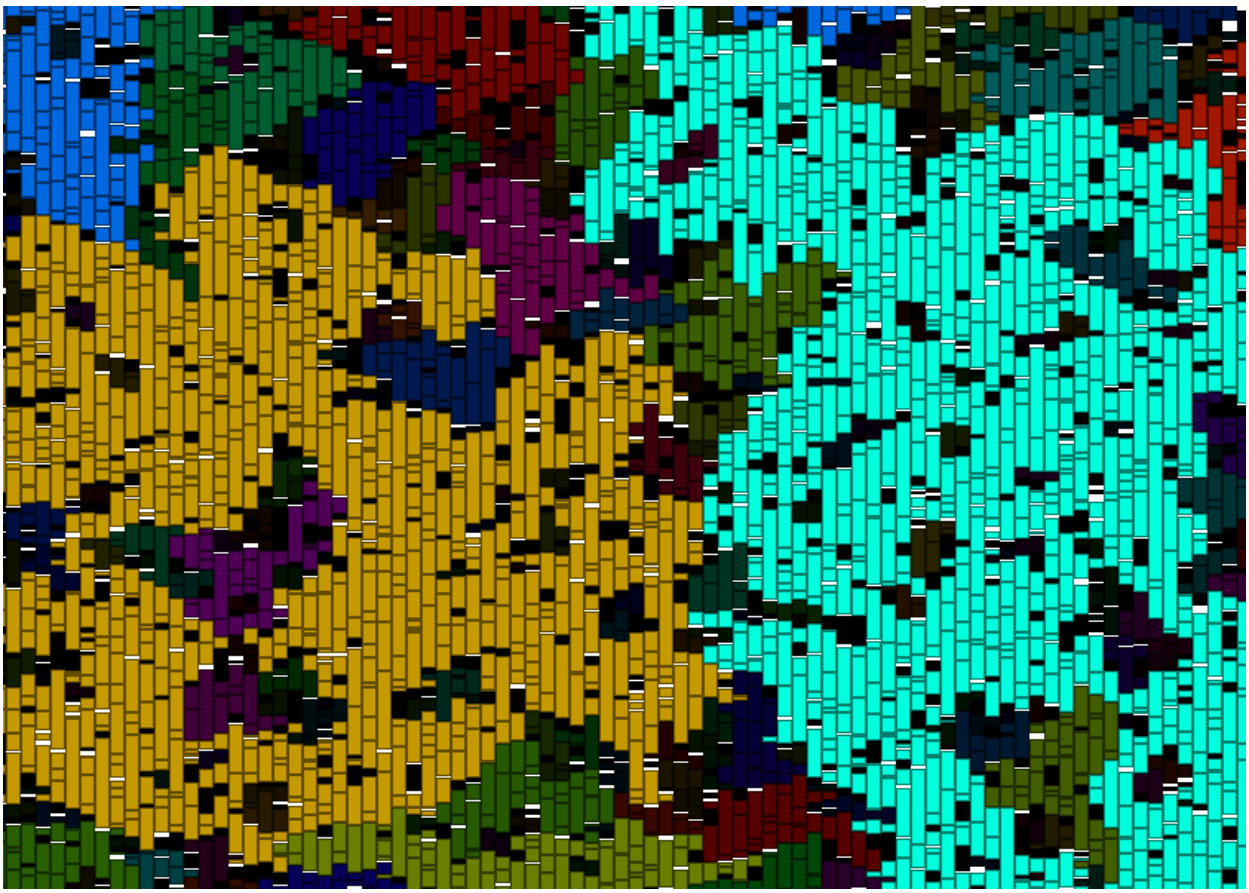


FIG. 4. (Color online) A sample realization of tube-based model with clusters of connected tubes marked by a single color: the brighter the color the larger the cluster size.

for which connection between lattice edges appears;

- $\hat{\rho}_{\rm av}$, $\hat{p}_{\rm av}$, $\hat{\mu}_{\rm av}$ - estimator of critical point for finite lattice, a mean value of, respectively, $\rho_{\rm con,k}$, and $\mu_{\rm con,k}$;
- ρ_i, p_i, μ_i - randomly generated value describing for which value of model parameter i-th bond is open.

A. Percolation threshold

For any percolation model on a square lattice $L \times L$ one defines the crossing probability $\Pi(\rho, L)$ as the probability that there is a connection between opposite edges (or faces, in 3D $L \times L \times L$ the case). The definition of a "connection" is model-dependent. For example, in the bond percolation model, it is a connected path, consisting of open bonds. In two

dimensions there are two such probabilities; we will refer to them as NS (north-south) and WE (west-east). In three dimensions there is a third possible direction of connection, which we call top-bottom (TB). The crossing probability depends on the size of the lattice and on the discrete-continuous model parameter ρ. In bond percolation models, ρ can be chosen as the open bond density p, but in general it need not represent a probability and its value need not lie in the interval $[0, 1]$. In the limit $L \to \infty$, Π converges to 0 for $\rho < \rho_c$ and to 1 when $\rho > \rho_c$. The critical value ρ_c is called the percolation threshold or the critical point, and depends on the type of lattice (e.g. square, triangular, etc. [2]). For a finite lattice, the transition is not sharp and many approximations of the critical point are used. Examples are the point p_{c1}, where the crossing probability is equal to 0.5 [1, 2], the point where the slope of Π (as a function of ρ) is largest, or, as used in this paper,

$$\rho_{av}(L) = \int \rho \frac{d\Pi}{d\rho}(\rho, L) d\rho. \tag{3}$$

The factor $\frac{d\Pi}{d\rho}(\rho, L)d\rho$ can be interpreted as the probability that the graph begins to percolate for a value of the model parameter in the interval $(\rho, \rho + d\rho)$. Thus ρ_{av} is the expected value of ρ at the onset of percolation [2]. Similarly, a measure of the width of the transition region can be defined as the variance

$$\Delta^2(L) = \int (\rho - \rho_{av})^2 \ \frac{d\Pi}{d\rho}(\rho, L) d\rho. \tag{4}$$

These quantities satisfy the scaling relations [2]

$$\rho_{av} - \rho_c \propto L^{-\frac{1}{\nu}} \tag{5a}$$

$$\Delta \propto L^{-\frac{1}{\nu}}, \tag{5b}$$

where ν is the (universal) critical length exponent. For additional results on finite-volume percolation thresholds, see [27, 28]. This leads to an asymptotic linear relation between $\rho_{av}(L)$ and $\Delta(L)$

$$\rho_{av} = a\Delta + \rho_c, \tag{6}$$

where a is a proportionality constant. Equation (6) provides a simple method of extrapolating results obtained for finite lattices to the infinite one. It is worth noticing that the scaling relations (3) - (6) are a discrete-continuous versions of the relations well known from the discrete percolation models.

B. Algorithms for the homogeneous lattice

We first present the simplest, most direct method of computing the critical density, a modification of which will be used in our algorithm. We consider the bond percolation model on a regular, homogeneous lattice. In this case, the discrete percolation model parameter (bond probability) p is identified with ρ from the previous section. We begin by assigning to each bond i a random number r_i, sampled from the uniform distribution on the interval $[0, 1]$. To simulate a realization with density p, we open the bonds for which $r_i \leq p$. We then check for existence of an open connection between the opposite sides of the lattice. Applying this with different p (for the same realization of the r_i), we find the value of p at which the connection first forms for a given realization p_{con}. The consecutive values of p are selected by a binary search. Repeating the whole procedure K times for different sets of random numbers r_i, we obtain a set of approximate values of $p_{\mathrm{con,k}}$ for k in $(1, 2, \ldots, K)$. This allows us to estimate p_{av} by the empirical mean value of p_{con}, with empirical variance Δ.

We take a different approach, similar to one taken in NZ. We compute the value of p_{con} at which an open connection appears for a given realization in a single run. However, instead of the 'microcanonical ensemble' used in NZ, we use a 'canonical ensemble'. The main advantage of our approach is its applicability to more general graphs, where probabilities vary from bond to bond. We note that the transformation between 'microcanonical' and 'canonical' ensemble representations is complicated and impractical in implementation. However, from the point of view of computing the percolation threshold on a homogeneous lattice, the two algorithms are equivalent, as explained in detail below. Furthermore, our algorithm shares with NZ the important feature of efficiently simulating many values of the control parameter in a single run.

Our approach to simulating multiple parameter values in a single run, inspired by a remark in [1], begins by assigning a random number to every bond, just as in the naive algorithm described above. To determine p_{con} (the empirical value of p at which percolation sets in) consecutive bonds i are opened in order of increasing r_i. We refer to this procedure as "rising water" because, just as one observes rising water covering objects in order of their height, so the bonds open according to their "height" as the value of p rises. Assuming that random numbers assigned to different bonds are different, at each stage we obtain the same

graph as when using the simplest method described above with $p = r_i$. The algorithm stops when a connection linking a fixed pair of opposite sides of the square is established. The estimate of $p_{\rm con}$ is equal to the value r_i of the last added bond. The results of applying the two algorithms are identical.

Indeed, suppose p_{cross} is the empirical crossing threshold determined by the rising water algorithm. Then the naive algorithm with the same sequence of r_i will also find a connection for any $p \geq p_{cross}$. On the other hand, for $p \leq p_{cross}$ no connection will be found, which shows that the two algorithms indeed yield the same result. The procedure described above is repeated K times, giving a set of results $p_{\rm con,k}$ for $k = 1, 2, \ldots, K$. The estimator of $p_{\rm av}$ is $\hat{p}_{\rm av} = 1/K \sum_1^K p_{\rm con,k}$.

C. Extension of the algorithm for discrete-continuous percolation

The procedure described above must be modified for more general models in which probabilities of connections depend on both the geometry and the model parameter. In the simplest algorithm applied to the tube-based model we have to generate random values r_i for all pairs of adjacent tubes and connect a pair of tubes with an overlap h_i when the condition

$$r_i \geq e^{-\mu h_i} \tag{7}$$

is fulfilled. Thus, we must compute, for every bond i, the smallest value of model parameter μ for which (7) is satisfied. We denote this value μ_i, thus

$$\mu_i = -\frac{\ln r_i}{h_i}. \tag{8}$$

When $\mu < \mu_i$ i-th bond is closed, and when $\mu \geq \mu_i$, it is open. Then, as in the homogeneous case, we sort the set of μ_i in increasing order and we open the bonds in the graph in this order. We estimate the critical value of parameter called $\mu_{\rm con}$ by the first value of μ_i at which a crossing between two fixed opposite sides of a square forms. In the most general case, the probability that i-th bond is open is given by relation $p_{{\rm bond}\,i} = F_i(\rho)$ where $F_i(\rho)$ are increasing functions mapping ρ to $[0, 1]$, and each of $F_i(\rho)$ can be a different function. The value of the model parameter at which the i-th bond is open in a given realization is

$$\rho_i = F_i^{-1}(r_i). \tag{9}$$

In Table I we compare our discrete-continuous algorithm with NZ.

The important parts of these algorithms are two main operations:

- finding the cluster containing a given site.
- connecting two clusters.

The use of union-find [8, 29, 30] algorithms for cluster identification in percolation dates back to the first Hoshen-Kopelman algorithm [22]. However, we follow Newman and Ziff by using an algorithm that is, for practical purposes, optimal. The "union-find" (or "disjoint-set") data structure stores information about connections in the form of trees where every site points either to another site from the same cluster, or to itself. The element pointing to itself, is the root of the tree and provides the cluster's identification. To find the cluster containing a given site, we follow the path indicated by the pointers until we reach the root. If for two sites we get the same root, both sites belong to the same cluster. To connect two different clusters we add a pointer between their roots. The most efficient union-find algorithms have two important features. The first one is to always point from the smaller tree to the bigger one ("balancing"). It requires storing the information about each cluster's size. The second is called path compression: having found the root of an element's cluster, we re-track the path from the element to the root again, changing the parent of each site along the way to the root. Using a union-find data structure makes operations of adding an edge and checking whether two sites belong to the same cluster very fast. In addition to the pointer to the parent and size of the subtree, one can store additional information in each site's record, such as moment of inertia, position, or the information about the cluster's connection to boundaries. The last one is a simple way to check whether the opposite parts of the boundary are connected. The position of a site can be used to check whether the cluster is wrapped around the torus. The amortized computational cost of using it is proportional to the inverse Ackermann function and thus it can be considered as a small constant for practical purposes [30].

D. Percolation statistics

As shown in Table I (step 4), an important property that our algorithm shares with NZ is its ability to simultaneously calculate functions of the bond variables of a given configuration

for different values of the model parameter ρ. While standard methods need K runs of the algorithm to compute K values of a model characteristic for a given set of model parameters ρ_k $(k = 1, \ldots, K)$, in our approach, all values are obtained simultaneously in a single run. Using union-find and rising water we efficiently obtain many important characteristics of the model, for example average cluster size, average moment of inertia and so on, with constant computational cost in every run of the algorithm. Other parameters, such as the histogram of cluster-size distribution with B bins, can be calculated with an additional cost proportional to the number of points in the realization (N) and to the number of bins. We now point out the major difference between our algorithm and NZ. Let us consider a quantity Q. In NZ we calculate $Q[i]$ which is the value of Q after adding the i-th bond. The values $Q[i]$ are then averaged over K different realizations, where the value of K depends on the required accuracy. As the next step we transform the result to the canonical value $Q(\rho)$ using

$$Q(p) = \sum_{n=0}^{N} \binom{N}{n} p^n (1-p)^{N-n} Q[n]. \tag{11}$$

In our algorithm we calculate $Q[j]$, the values of Q for a chosen collection of values of model parameters ρ_j and, in order to estimate Q at the critical threshold, we take the $Q[j]$ obtained in the last step of the algorithm (described in Table I as a step 4) for which we had $\rho < \rho_i$. But, if desired we can also collect statistics for other values of ρ as we "raise the water". The efficiency of our algorithm in doing so relies on fast updates of Q, using operations on clusters rather than having to run through the whole graph at each step. For example, we consider the cluster size. The size s_C of cluster C obtained as a union of two clusters A and B is equal to

$$s_C = s_A + s_B. \tag{12}$$

Similarly, for the calculation of the moment of inertia for clusters we use the stored quantities: sizes of clusters s_i, masses of clusters m_i, centers of mass r_i and previous moments of inertia I_i. For unions of clusters we obtain:

$$m_c = m_a + m_b \tag{13a}$$

$$r_c = \frac{r_a m_a + r_b m_b}{m_c} \tag{13b}$$

$$I_c = I_a + I_b + (r_a - r_c)^2 m_a + (r_b - r_c)^2 m_b \tag{13c}$$

Note that (13c) is the parallel axis theorem (Steiner law). In our method, if we store in memory information about the clusters, all these operations have only a constant cost per operation. For example to get a mean value of the moment of inertia we additionally store in memory the sum of the moments of inertia of the clusters and update this sum.

An important application of the above method is to compute the wrapping probability $R_L(p)$, which is the probability that a cluster wraps around a lattice with periodic boundary conditions. This probability may be used to approximate p_c by the value of p for which $R_L(p)$ is equal to $R_\infty(p_c)$. The last quantity is known exactly in two dimensions [31]. Denoting the approximate value for the lattice of size L by p_c^L we have the scaling relation $p_c - p_c^L \sim L^{-2-\frac{1}{\nu}}$. In a vicinity of the critical point, the wrapping probabilities are regular functions and thus the above procedure yields an efficient estimate. In order to compute the wrapping or spanning probability (or, more generally, the probability of any *increasing event* [1]) we just need to record the unique value of $\rho_{\mathrm{con,k}}$ at which the event appears. Sorting these values we obtain an empirical distribution function which gives the probability of the desired event:

$$P[\rho] = \frac{\sharp\{\rho_{\mathrm{con,k}} \leq \rho\}}{\sharp\{\rho_{\mathrm{con,k}}\}} \tag{14}$$

E. Critical exponents

When the percolation threshold ρ_{c} (μ_{c} in tube-based model) is computed, a postprocessing algorithm gathers statistics about the distribution of clusters (including the size of the largest cluster, cluster-size moments, cluster-volume moments). These statistics are determined for ρ in a vicinity of ρ_{c}. This allows computing several critical exponents of the model. In particular the cluster-size distribution near the percolation threshold allows computing the Fisher exponent τ. The exponent β is computed from the size of the maximal cluster. From data acquired in the algorithm outlined in Sec. III C exponent ν in (5b) can be computed using the scaling relation (5b).

IV. RESULTS IN THE TWO-DIMENSIONAL CASE

A. Percolation threshold

The simulation was run for several square lattices with size ranging from $L = 200$ to $L = 10000$. The estimators of μ_{av} and Δ were acquired for mutually perpendicular directions, denoted by NS (North to South) and WE (West to East or left to right). The percolation threshold for the infinite lattice ($L \to \infty$) was computed by fitting the data to the scaling properties described by (6) as presented in Fig. 5. The results for the infinite lattice based on the intercept of the fitted linear function are the following:

$$\mu_{\text{c NS}} = 0.99999 \pm 2.5 \times 10^{-5} \tag{15a}$$

$$\mu_{\text{c WE}} = 0.99999 \pm 5.0 \times 10^{-5} \tag{15b}$$

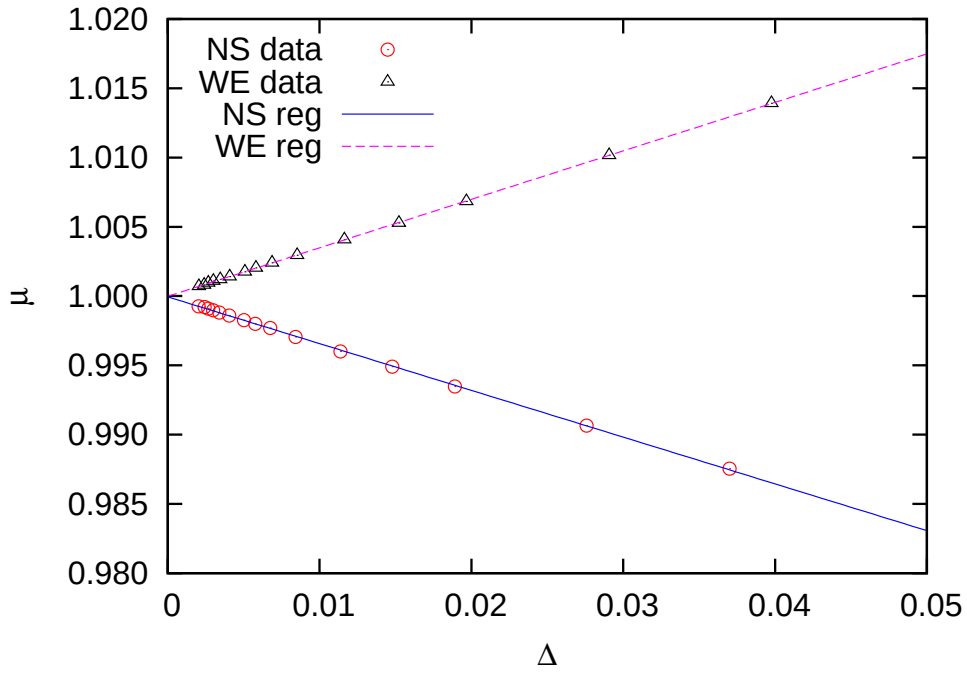
It is worth noting that values μ_{av} converge to μ_{c} from both directions, as presented in Fig. 5. The obtained value of μ_{c} equal 1 is clearly model-specific, as discussed in Sec. IV B.

B. Duality and exact analytic calculation of μ_c

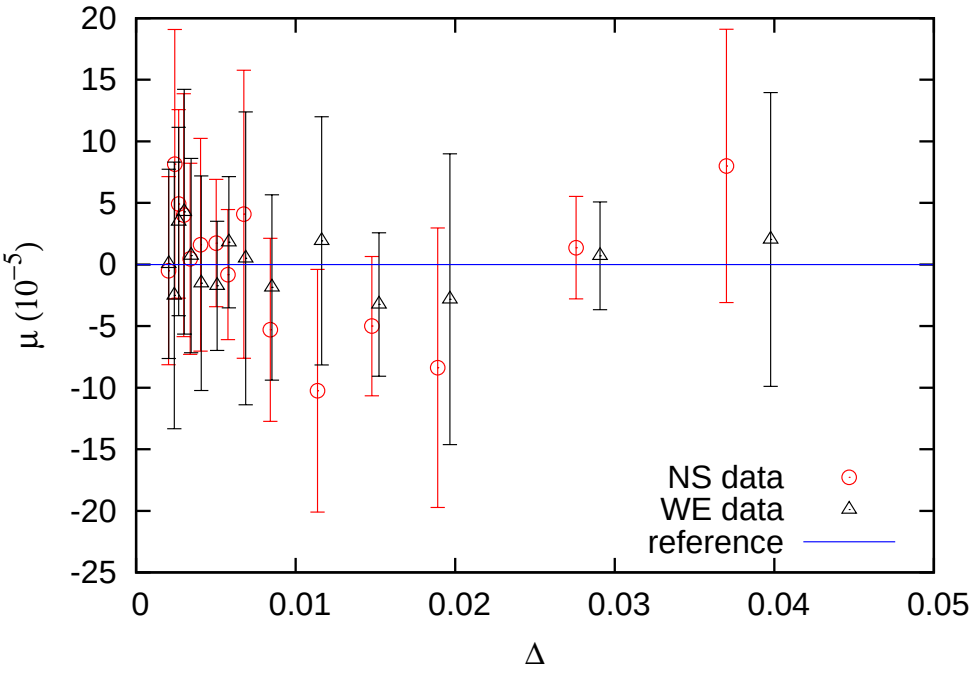
We consider a realization of the two dimensional graph defined by $\{L, n, \mu_1, \mu_2\}$ presented in Fig. 6a. We define the graph dual to the initial one according to the following procedure:

- dual lines are introduced, each line is placed between two existing lines;
- dual lines are divided into tubes (dual tubes) by the bonds of initial graph (vertical segments marked in Fig. 6b;
- at the break points between initial tubes the dual bonds connecting dual tubes are introduced (horizontal lines marked in Fig. 6b.

The two graphs, initial and dual, are shown in Fig 6a and c. New tubes and bonds are distributed in the same way as the original ones, with the two Poisson process parameters interchanged. Notice that the two graphs have no intersections. We either have a connection from North to South, using tubes and bonds of the original graph, or we can draw a line through the empty spaces and breaks between the tubes from West to East, that does not cross any bonds or tubes. In the latter case, there is a connection from West to East in the

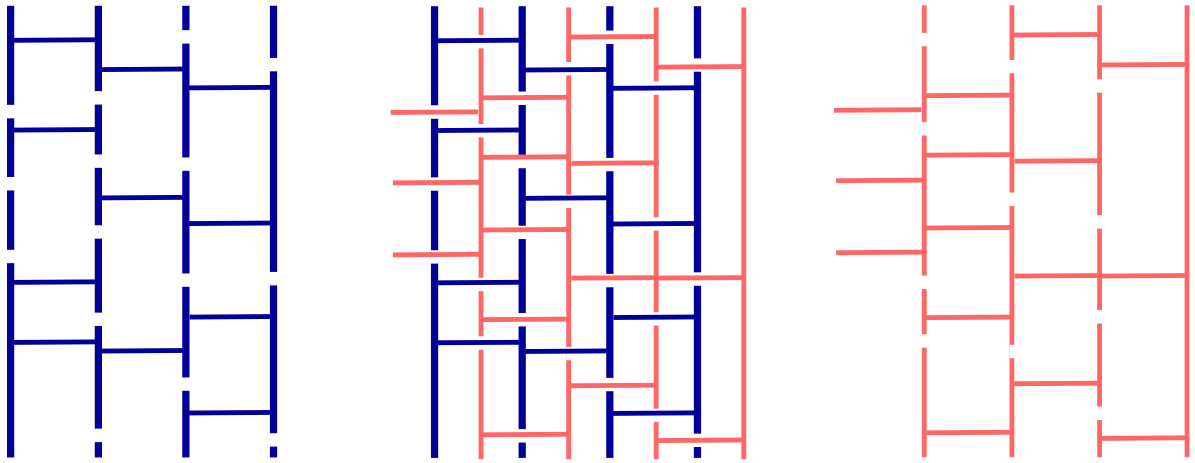


(a)



(b)

FIG. 5. (Color online) (a) The percolation threshold computed by studying North-to-South (green points) and West-to-East (red points) connections. The statistical uncertainty associated with data points is smaller than their size. In (b) the differences between data points and the fitted line are shown.



(a) (b) (c)

FIG. 6. (Color online) Construction of the dual graph from the original one: (a) the original graph, (b) the construction and (c) the final dual graph.

dual graph. Similarly, exactly one of the two alternatives occurs: either there is a connection from West to East by bonds and tubes of the original graph, or there is a connection from North to South in the dual graph— an unbroken path through empty spaces. A given realization of the original graph starts to percolate when the dual graph stops percolating, so $\mu_{\rm av} = \mu_{\rm av}^{\rm dual}$ for a pair of dual graphs. We know that the percolation threshold in the limit $n = L \to \infty$ depends only on the ratio μ_2/μ_1. Increasing μ_1 results in more (shorter) tubes and thus makes percolation more difficult, while increasing μ_2 makes for more connections between tubes, which facilitates it. Together with the duality described above, this indicates that $\frac{\mu_2}{\mu_1} = 1$, i.e. $\mu_c = 1$ is the percolation threshold, thus explaining the numerical result (15a) and (15), and giving further support to our method. We emphasize that a rigorous proof that the critical value of μ equals 1 requires a more careful argument. The first result of this type (for the square lattice) was proven in [32]. Simpler arguments developed later can be found in [1]. They can be adapted to cover the present case as well.

C. Critical exponents

Based on the scaling laws in (5a) and (5b) we obtain the correlation length exponent: $\nu = 1.345 \pm 0.009$. The exact value is known to be 4/3. Moreover, the correctness of the result indicates that relations (5a), (5b) and (6), originally defined for discrete percolation, are valid also in case of discrete-continuous percolation. In discrete-continuous percolation the size of a cluster can be characterized in two ways. In the first way, like in the discrete model, we count the number of sites (tubes) in the cluster. The total number of sites defines a cluster size. Secondly, we sum the lengths of tubes in the cluster. This value defined cluster volume. The distributions of these two characteristics are presented in Fig. 7a, and Fig. 7b.

The Fisher exponent τ is determined from the cluster size distribution presented in Fig. 7a, as $\tau = 2.046 \pm 0.023$. The exact value is $187/91 \approx 2.054$ [2]. This agreement of the results with the known values of critical exponents supports the validity of the algorithm. Moreover, we show that the slopes of lines fitted in Figs. 7a and b are the same, thus the values of the Fisher exponent based on cluster size distribution and on cluster volume distribution are the same.

V. RESULTS IN THREE DIMENSIONS

The simulation was run for cubic lattices with linear size ranging from $L = 100$ to $L = 400$. The estimators of ρ_{av} and Δ were acquired for perpendicular directions, denoted by NS, WE and TB (top to bottom). As in Sec. IV we used (5b) to compute the percolation threshold for infinite lattice. The results are as follows:

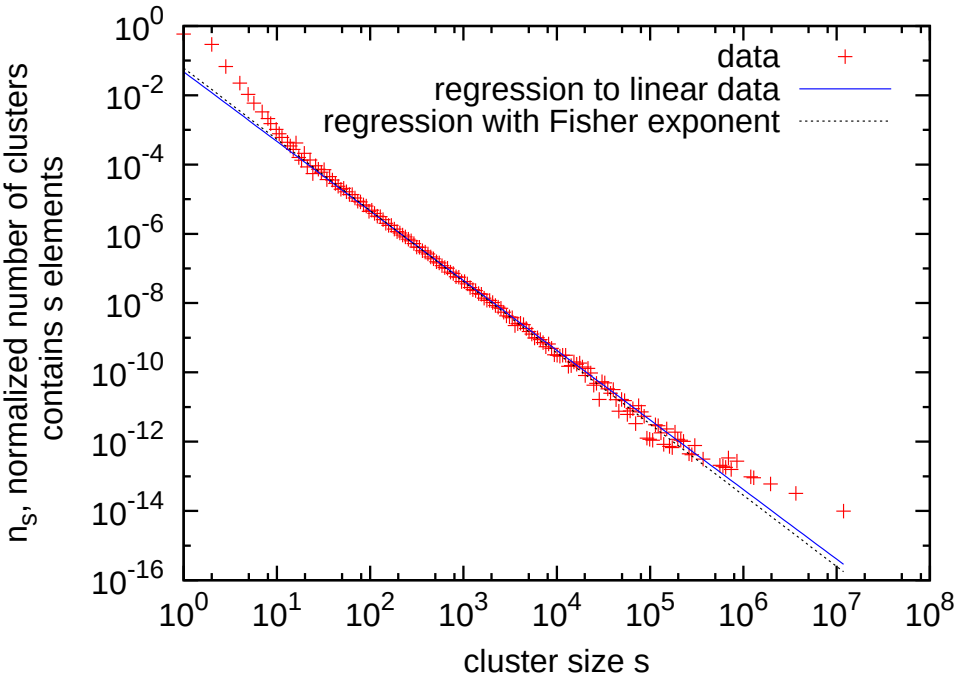
$$\mu_{\text{c NS}} = 0.231466 \pm 6 \times 10^{-6} \tag{16a}$$

$$\mu_{\text{c WE}} = 0.23146 \pm 7 \times 10^{-6} \tag{16b}$$

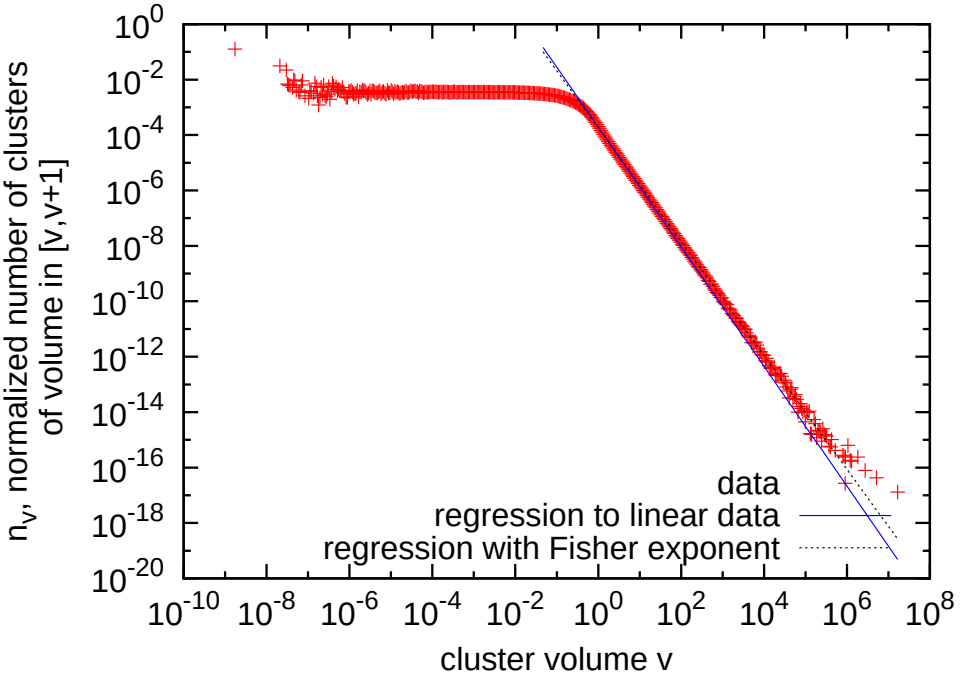
$$\mu_{\text{c TB}} = 0.23140 \pm 1.2 \times 10^{-5} \tag{16c}$$

The results obtained by fitting independently three linear functions, as presented in Fig. 8a, can be improved using the following constraints:

- the lines fitted to the results perpendicular to tubes (NS and WE) have the same slope and intercept b;



(a)



(b)

FIG. 7. (Color online) Number of clusters: (a) n_s - number of clusters of size s per one site. (b) n_v - number of clusters of volume v per unit of volume. Data from 2D grid 15000×15000.

- the line fitted to the results parallel to tubes (TB) have also the same intercept b.

Thus the improved estimated value of the percolation threshold is:

$$\mu_c = 0.231456 \pm 6 \times 10^{-6} \tag{17a}$$

It is worth noting that, as in the two-dimensional case, which we discussed in Sec. IV B, the values μ_{av} converge to μ_c from both directions. This is clearly visible in Fig. 8.

VI. DISCUSSION

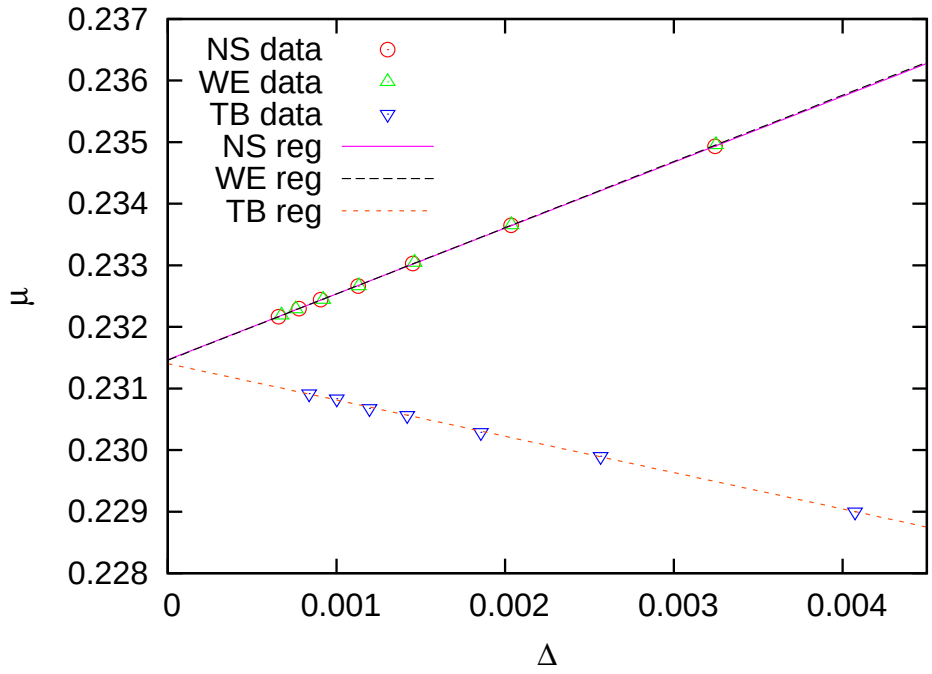
A. Computational costs

The three main parts of the algorithm increase computational costs:

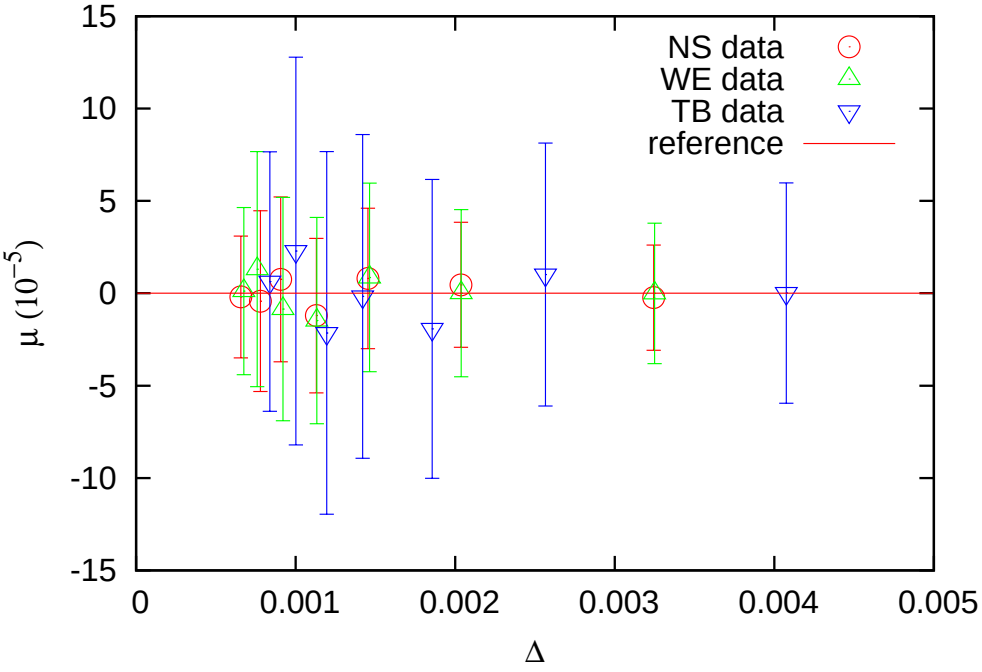
- creating the connections (witch includes random number generation),
- sorting them,
- running 'union' for every connection.

These three parts have similar run-time costs, each one contributing 10-50 % of the total simulation time, depending on the size of the simulations. In the 3D case last part is the fastest, taking 10-15% of the total running time. For a system with N bonds, both the presented algorithm and the NZ algorithm require generating N random numbers (more precisely, $N-1$ in the NZ case). While in the NZ algorithm the random numbers are used to generate a random permutation of connections, in our case they determine the values of p_i or μ_i at which the bonds become open as described in Eq. (8). In addition, while the NZ algorithm uses the uniform distribution, which can be generated directly, we have to generate other distributions. This requires computing functions of uniformly distributed random numbers. In the activated carbon model, we use the logarithmic function. Note, that in the case of a standard discrete percolation model on a uniform lattice the uniform distribution is used, and thus no additional computation is necessary. As opposed to the NZ algorithm, in order to treat inhomogeneous lattices, our algorithm contains an additional sorting component.

The memory requirement is linear in the size of the graph, like in the NZ algorithm. The Hoshen-Kopelman algorithm, for instance, is better suited for simulating extremely



(a)



(b)

FIG. 8. (Color online) (a) The percolation threshold computed from top to bottom connections (green points) and from left to right connections (red points). In (b) the differences between data points and the fitted lines are shown.

large lattices for a single value of ρ, because, in this case, its memory requirement scales as $O(L^{d-1})$ but the run time would be comparable to NZ or our algorithm.

B. Computational cost versus accuracy of results

The computational cost of determining an approximate value of the critical point ρ_{av} depends on the size of the lattice and on the desired accuracy of calculation, which can be expressed in terms of standard deviation $\Delta(L)$. Analysis of this computational complexity allows us to know what accuracy ϵ can be achieved in a given time. The obtained value of ρ_{av} is approximated by the Monte Carlo estimator $\hat{\rho}_{\mathrm{av}}$, which takes into account all runs of the algorithm:

$$\Delta\hat{\rho}_{\mathrm{av}} \propto \frac{\sigma_{\hat{\rho}_{\mathrm{av}}}}{\sqrt{K}} \tag{18}$$

where K denotes the number of repetitions of the rising water algorithm (the second point in the Table I for our algorithm). Thus ϵ, the final accuracy of $\hat{\rho}_{\mathrm{av}}$, depends on the number of runs of the algorithm and on the variance $\Delta(L)$ as follows:

$$\epsilon = \frac{\Delta(L)}{\sqrt{K}} \tag{19}$$

From (5b) we know that $\Delta(L)$ depends on the size of the domain L. The computational cost $c = KL^d \log n$ is proportional to the number K of times the main loop of the algorithm is repeated and to the cost of a single run (of the order of $L^d \log n$, where d is the dimensionality of the problem). Thus the computational cost to obtain the result with the accuracy ϵ is

$$c = \frac{L^{d-\frac{2}{\nu}} \log n}{\epsilon^2}. \tag{20}$$

The exponent $d - \frac{2}{\nu}$ depends on the dimension of the problem. For the two-dimensional case it is $1/2$, while in three dimensions it equals approximately 0.72. The logarithmic factor in the expression for the computational cost (20) is due to sorting of random numbers in step 2c in Sec. I. One method to avoid this is to use the so-called bucket sort, which is a linear-time sorting algorithm using information about data distribution [30]. Due to statistical behavior of the random values ρ_i we can create a set of disjoint intervals that cover all possible values of ρ_i and have approximately the same expected number of random values ρ_i in each interval. Let us denote this expected number of random variables in one interval ("bucket") by M. For every generated random ρ_i $(i = 1, \ldots, N)$ we can compute in constant time to which

bucket it should be assigned. When all numbers are generated and classified, in each bucket we have a set of $M + O\left(\sqrt{M}\right)$ numbers, and we need to sort it. The computational cost of generating N random variables and sorting N/M buckets of size M is $O(N \log M)$ and it is linear in N because it is always possible to generate enough intervals to keep M constant. After that, the cost of running the algorithm K times is

$$O(KL^d \alpha(L^d))$$

and cost of running the algorithm to get desired accuracy ϵ is

$$O\left(\frac{L^{d-2/\nu}\alpha(L^d)}{\epsilon^2}\right)$$

Here $\alpha(L^d)$ is the inverse of Ackermann function and can be considered constant.

Despite better asymptotic behavior of the bucket sort, it does not give a better performance except for very big lattices.

C. Application of algorithm and discrete-percolation models

The presented algorithm has a much wider range of applicability—it can be used for any bond percolation model in which bonds are open with arbitrary (nonuniform) probabilities. Such models arise naturally in situations similar to the one studied here, in which realizations of the model are graphs, built from randomly generated geometric objects in a Euclidean space in which the probabilities of open bonds depend on the particular graph realization. With a choice of nanopipes as the main building blocks of the ordered part of carbon structure our model can be considered as belonging to the family of carbon nanotube (CNT) bundles models [33–35]. In the present work we limit ourselves to a relatively simple example, relevant for describing the internal structure of charcoal of activated carbon, but various aspects of CNT bundles theory indicate possible extensions of our work. Similarly to what is done in analysis of the properties of the carbon nanotubes immersed in various resins [36], we can allow nanotubes which are not fully aligned [37]. In addition, we can allow their electrical properties to vary, part of them being metallic and part semiconducting [38]. Suitable modifications of the presented algorithm can be applied to these and other generalizations.

It is worth mentioning that the problem of quantum aspects of the carbon activation process is to a great extent open. This suggests to study quantum versions of the family

of the discrete-continuous models discussed in this paper. The interplay of discrete and continuous aspects may lead to quantitatively novel effects. We note that such quantum disordered models can in principle be *quantum simulated* by a system of ultracold atoms (see, for instance, Ref. [39]): an array of random length 1D Bose condensed gases with controlled random connections between them.

VII. CONCLUSIONS

In summary, three goals have been achieved in this work:

- We have defined percolation models motivated by the physics of activated carbon. These models deal with tubes of random length connected by random bonds; as such they describe well situations in which complicated micro-structures observed in activated carbon have approximately linear textures: carbon nanotubes or graphene ribbons. In cases when the structures are neither 1D nor quasi-1D (fullerenes [18, 19]), stacked graphite [20], or graphene [15], carbon onions [21]), the concrete models considered here provide only a "caricature" of the real situation, capturing only qualitative aspects of the underlying physics.

- We have generalized the above model and have defined a family of discrete-continuous percolation models extending standard discrete percolation to lattices with inhomogeneous connection probability.

- We have presented an efficient algorithm for treating inhomogeneous lattices, especially including the discrete-continuous family. We have analyzed in detail its computational costs and found that they are similar to those of the efficient Newman-Ziff algorithm.

- We applied the extended algorithm to the family of models in question, calculating critical values of the model parameters, critical exponents and cluster density distributions in two and three dimensions.

Possibilities for further studies include: i) applications of the present models to experimental data, suggesting geometry formed by parallel random tubes/ribbons connected randomly by bonds; ii) development of concrete models with geometry formed by parallel random

flakes/patches connected randomly by bonds; iii) application of the method to such models, calculation of their properties, and direct comparison with experiments.

ACKNOWLEDGMENTS

We are grateful to R. Ziff for insightful comments on the first version of the paper. This work has been partially supported by the Iuventus Plus program founded by the Polish Ministry of Science and Higher Education (IP2014 024373). K.K. acknowledges START Programme founded by the Foundation for Polish Science (START 063.2014). M.L. acknowledges Spanish MINECO Projects (FOQUS FIS2013-46768 and SEVERO OCHOA GRANT SEV-2015-0522), ERC AdG OSYRIS, EU IP SIQS, EU STREP EQuaM, and EU FETPROACT QUIC. J.W. has been partially funded by NSF grant DMS 131271 and a part of his work on this article was supported by NSF under grant DMS 1440140 while he was in residence at the Mathematical Sciences Research Institute in Berkeley during the Fall of 2015 semester.

[1] G. R. Grimmett, *Percolation (Grundlehren der mathematischen Wissenschaften)* (Springer: Berlin, Germany, 2010).

[2] A. Aharony and D. Stauffer, *Introduction to percolation theory* (Taylor & Francis, United Kingdom, 2003).

[3] R. Meester and R. Roy, *Continuum percolation*, Cambridge Tracts in Mathematics, Vol. 119 (Cambridge University Press, Cambridge, 1996).

[4] Z. Wu, L. A. Braunstein, S. Havlin, and H. E. Stanley, Phys. Rev. Lett. **96**, 148702 (2006).

[5] D. Li, B. Fu, Y. Wang, G. Lu, Y. Berezin, H. E. Stanley, and S. Havlin, Proceedings of the National Academy of Sciences **112**, 669 (2015).

[6] M. Franceschetti, L. Booth, M. Cook, R. Meester, and J. Bruck, Journal of Statistical Physics **118**, 721 (2005).

[7] L. Booth, J. Bruck, M. Franceschetti, and R. Meester, The Annals of Applied Probability **13**, pp. 722 (2003).

[8] Z. Galil and G. F. Italiano, ACM Comput. Surv. **23**, 319 (1991).

[9] M. E. J. Newman and R. M. Ziff, Physical Review E **64**, 016706 (2001).

[10] S. Furmaniak, A. P. Terzyk, P. A. Gauden, P. Kowalczyk, and P. J. Harris, Journal of Physics: Condensed Matter **26**, 485006 (2014).

[11] S. Furmaniak, A. P. Terzyk, P. A. Gauden, and P. Kowalczyk, Microporous and Mesoporous Materials **154**, 51 (2012).

[12] H. Marsh, *Activated carbon compendium: a collection of papers from the journal carbon 1996-2000* (Gulf Professional Publishing, 2001).

[13] B. Feng and S. K. Bhatia, Energy and Fuels **14**, 297 (2000).

[14] K. Kwiatkowski, K. Bajer, A. Celińska, M. Dudyński, J. Korotko, and M. Sosnowska, Fuel **132**, 125 (2014).

[15] M. Pawlyta, Materials Science and Engineering **63**, 58 (2013).

[16] P. J. Harris, Journal of Materials Science **48**, 565 (2013).

[17] X. Wang, G. Sun, and P. Chen, Frontiers in Energy Research **2**, 33 (2014).

[18] P. J. Harris, Chemistry and physics of carbon **28** (2003).

[19] H. Marsh and F. R. Reinoso, *Activated carbon* (Elsevier, 2006).

[20] G. Jenkins and K. Kawamura, Nature **231**, 175 (1971).

[21] Y. Chen, C. Liu, F. Li, and H.-M. Cheng, Journal of Porous Materials **13**, 141 (2006).

[22] J. Hoshen and R. Kopelman, Physical Review B **14**, 3438 (1976).

[23] J. Hoshen, P. Klymko, and R. Kopelman, Journal of Statistical Physics **21**, 583 (1979).

[24] J. Hoshen, M. W. Berry, and K. S. Minser, Physical Review E **56**, 1455 (1997).

[25] Z. Alexandrowicz, Physics Letters A **80**, 284 (1980).

[26] P. Leath, Physical Review B **14**, 5046 (1976).

[27] L. Berlyand and J. Wehr, Communications in Mathematical Physics **185**, 73 (1997).

[28] L. Berlyand and J. Wehr, Journal of Physics A: Mathematical and General **28**, 7127 (1995).

[29] B. A. Galler and M. J. Fisher, Comm. ACM **7**, 1963 (1964).

[30] T. H. Cormen, *Introduction to algorithms* (MIT press, 2009).

[31] H. T. Pinson, Journal of statistical physics **75**, 1167 (1994).

[32] H. Kesten, Communications in Mathematical Physics **74**, 41 (1980).

[33] H. G. Chae and S. Kumar, Science (2008).

[34] N. Behabtu, C. C. Young, D. E. Tsentalovich, O. Kleinerman, X. Wang, A. W. Ma, E. A. Bengio, R. F. ter Waarbeek, J. J. de Jong, R. E. Hoogerwerf, *et al.*, Science **339**, 182 (2013).

[35] A. V. Kyrylyuk and P. van der Schoot, Proceedings of the National Academy of Sciences **105**, 8221 (2008).

[36] X. Zeng, X. Xu, P. M. Shenai, E. Kovalev, C. Baudot, N. Mathews, and Y. Zhao, The Journal of Physical Chemistry C **115**, 21685 (2011).

[37] F. Du, J. E. Fischer, and K. I. Winey, Physical Review B **72**, 121404 (2005).

[38] F. Xu, Z. Xu, and B. I. Yakobson, Physica A: Statistical Mechanics and its Applications **407**, 341 (2014).

[39] M. Lewenstein, A. Sanpera, and V. Ahufinger, *Ultracold atoms in optical lattices: Simulating quantum many-body systems* (Oxford University Press, Oxford, 2012).

Newman-Ziff	Discrete-continuous algorithm
1. create a table $Q[1:N]$ to store statistic	1. for a given set of values of model parameter p_l^s (where $l = 1, 2, \ldots$) create a table $Q[\ldots]$ to store statistic
2. run K times for k=1:K	2. run K times for k=1:K
a) create a list of all bonds	a) create a list of all bonds
b) generate a permutation of connections: j_i means that j-th bond will be added in i-th step	b) assign a random number r_i to every connection and compute value of model parameter p_i [μ_i from (8)] for which we add the bond. Sort connections in order of increasing p_i. Let j_i denote this sorting permutation
c) initialize the list of clusters so that each site is an a cluster of exactly one site	c) initialize the list of clusters so that each site is an a cluster of exactly one site
d) for i=1:N do	d) for i=1:N do
- look at bond j_i connecting sites a and b. If these sites belong to different clusters A and B, merge both clusters	- look at bond j_i connecting sites a and b. If these sites belong to different clusters A and B, merge both clusters
- check for spanning: for the first occurrence save iteration number i as i_k	- check for spanning, for the first occurrence save p_i number as $p_{\mathrm{con},k}$
- refresh the statistics in merged cluster and table $Q[i]$	- refresh the statistics in merged cluster and if for any i, $p_{i-1} \le p_l^s < p_i$, update the statistics $Q[p_l^s]$
3. compute the percolation threshold using the values of i_k	3. compute the percolation threshold $\hat{p}_{\mathrm{av}}$ and its variance $\hat{\Delta}_{\mathrm{av}}$ using $p_{\mathrm{con},k}$ as follows: $\hat{p}_{\mathrm{av}} = \frac{1}{K}\sum_{k=1}^{K} p_{\mathrm{con},k}$ and $\hat{\Delta}_{\mathrm{av}} = \sqrt{\frac{1}{K-1}\sum_{k=1}^{K}(p_{\mathrm{con},k} - \hat{p}_{\mathrm{av}})^2}$
4. compute the transformation from microcanonical $Q[n]$ to canonical $Q(p)$ using the following formula $$Q(p) = \sum_{n=0}^{N} \binom{N}{n} p^n (1-p)^{N-n} Q[n] \qquad (10)$$	

TABLE I. Comparison of the NZ algorithm to our discrete-continuous algorithm.

L	sites	bonds	time [s]	memory [MB]
50	130045	489934	0.11	10
70	352948	1352669	0.32	36
100	1020056	3959754	0.96	75
150	3420357	13410551	3.57	287
200	8079989	31838507	8.29	566

TABLE II. Mean number of sites (each one requiring one random number, sampled from the uniform distribution), mean number of bonds (each one requiring a random number sampled from the exponential distribution), mean running time necessary to compute single value of p_{con}, peak value of required memory from whole simulation, for a given lattice size L.