Statistical Geometry Described.

1. Introduction; An Example

A statistical geometry pattern is shown in Fig. 1. The essence of the algorithm is that circles are placed within a bounding circle (or other shape) in order of decreasing size, with a *random search* each time for a position where it does not overlap any previously placed circle. Available information indicates that this algorithm can be used *for any shape* (not just circles), and that under a wide range of parameters *it never halts* despite the random placement. It is *space-filling*.

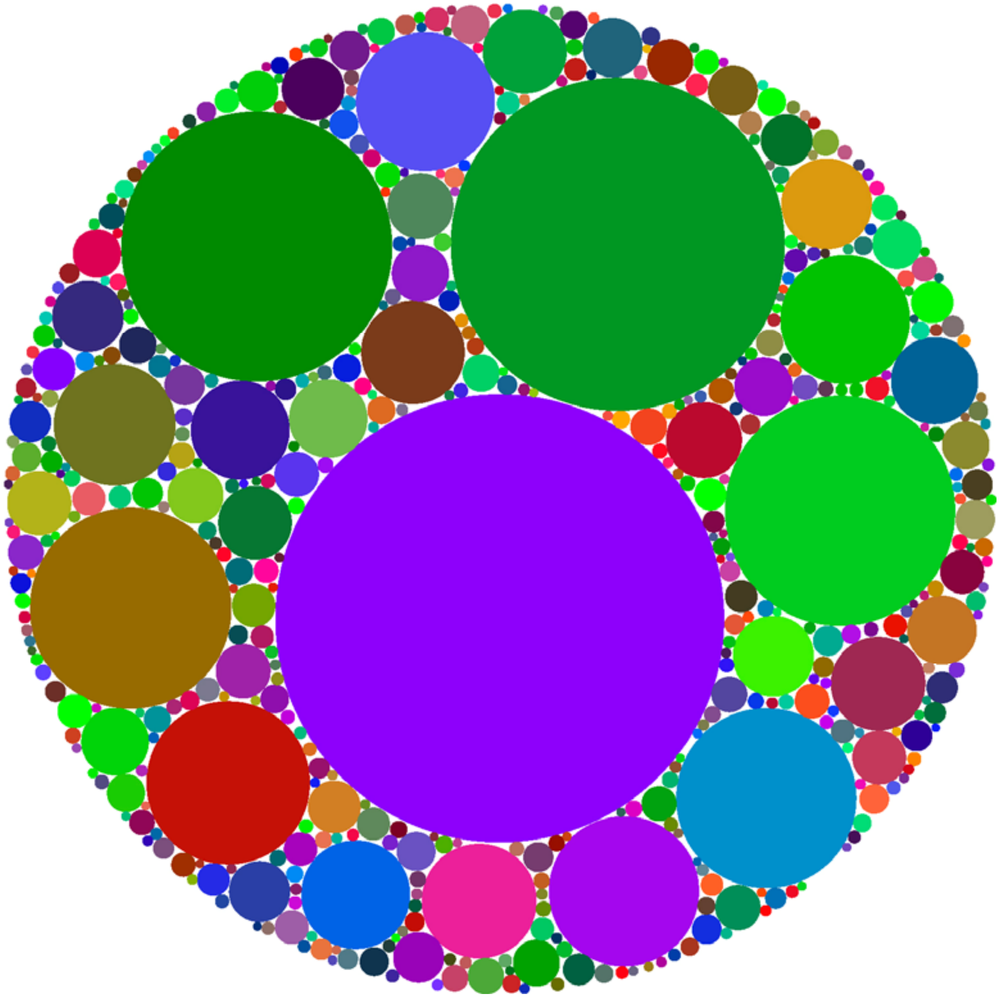


Fig. 1. A set of randomly placed circles. There are 500 circles, with 93% fill. Modified random color.

2. The Algorithm Stated.

Fractals have come to be accepted in mathematics, physics, and several fields of practical application since the ground-breaking book by Mandelbrot [1]. The fractals of interest here are generated using the following algorithmic rules:

1. Create a sequence of areas A_i equal to $\frac{1}{N^c}, \frac{1}{(N+1)^c}, \frac{1}{(N+2)^c}, \frac{1}{(N+3)^c}, \ldots$ where c and N are constant parameters. Choose an area (square, rectangle, circle, …) A to be filled.

2. Sum the areas A_i to infinity, using the Hurwitz zeta[1] function

$$\zeta(c,N) = \sum_{i=0}^{\infty} \frac{1}{(N+i)^c}$$

3. Define a new set of areas S_i by $S_i = \frac{A}{\zeta(c,N)(N+i)^c}$. It will be seen that the sum of all these redefined areas is just A.

4. Let $i = 0$. Place a shape having area S_0 in the area A at a random position x,y such that it falls entirely within area A. This is the "initial placement". Increment i.

5. Place a shape having area S_i entirely within A at a random position x,y such that it falls entirely within A. If this shape overlaps with any previously-placed shape repeat step 5. This is a "trial".

6. If this shape does not overlap with any previously-placed shape, store x,y and the shape dimensions in the "placed shapes" data base, increment i, and go to step 5. This is a "placement".

7. Stop when i reaches a set number, percentage filled area reaches a set value, or other.

The linear dimensions of the shapes are nowhere specified. They are calculated from the area for the shape of interest. A very wide variety of shapes have been found to be "fractalizable" in this way.

This is a very *simple* algorithm, easily stated in a few lines of text. While the algorithm is simple, it is able to produce complicated[2] results.

The parameters[3] c and N can have a variety of values. In practice the parameter c is usually in the range[4] 1.2-1.4 with a largest usable value around 1.51 for squares (with lower largest c values for hard-to fractalize[5] shapes). N can be 1 or larger, and need not be an integer.

[1] The definitions of the Riemann and Hurwitz zeta functions can be found in Wikipedia. The Riemann zeta function is historically older than the Hurwitz function, and is the special case where $N = 1$. In my images N is usually taken to be an integer, but according to the definition of $\zeta(c,N)$ it can be any real number ≥ 1. The Riemann zeta function has been much studied in connection with number theory, but it is not evident that such studies have any relevance here.

[2] This is a characteristic of many iterative or recursive procedures such as cellular automata and some fractals. Random walk, with its vast number of variations of both form and detail provides another example.

[3] The parameters used here describe any case where a pure power law is used. It is possible to fill the area A up to about 30-40% with almost any set of shapes, and the algorithm will still work if you go over to a power law at that point, taking A in the setup as the remaining unfilled area.

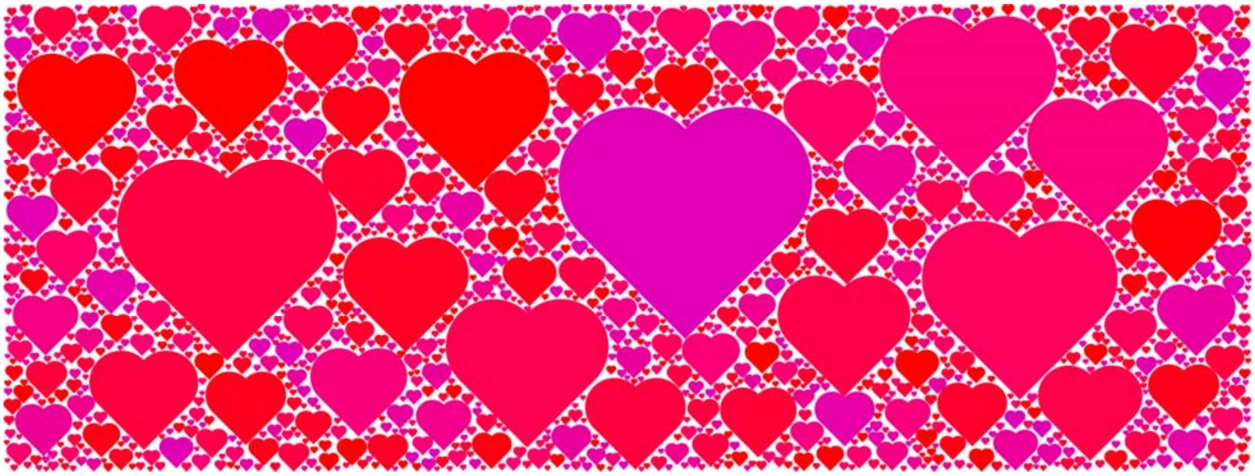


Fig. 2. Fractalized hearts. The algorithm, if continued *ad infinitum*, would eventually fill the entire space with ever-smaller hearts. 1500 hearts, c = 1.28, N = 4, fill = 82%. White gasket, inclusive boundaries. These hearts are constructed from a diamond and two semicircles.

By construction the result is a space-filling random fractal -- *if the process never halts*[6]. Available evidence (sec. 10, below) says that it does not halt, at least for c values which are not close to the upper limit of c. The power law area sequence ensures that it has the fractal "statistical self-similarity" (scale-free) property.

The random search ensures that no two circles will ever touch (with perfect-resolution numbers), so that the "gasket" is a single continuous object. This is not complete randomness, but *constrained* randomness. Each placement is constrained by the results of all the previous placements. The degree of constraint depends strongly on the parameter c [5].

There is no proof that a power law is the only and unique choice for area as a function of i. The dimensionless gasket width [5] offers some explanation of why the power law works so well.

Failure of the algorithm at high c appears to occur because the process becomes quite noisy and it will be seen that in such cases some runs continue indefinitely, while others fail because there is apparently no place big enough for the next shape. Such high-c failures usually occur during the first few placements.

[4] If $c < 1.2$ the number of trials needed to achieve a substantial percentage fill becomes huge. If $c > 1.4$ the number of trials per placement becomes huge.

[5] This is a new verb. Given the large variety of shapes that can be made into space-filling fractals by the statistical geometry algorithm it seems appropriate.

[6] There are two ways to consider the halting question. One can ask whether the *computational algorithm* using floating-point numbers halts. Here the answer is probably yes, since there is a smallest feature size of 1 least-significant bit. Or one can ask whether the algorithm halts for a mathematician's numbers, which have infinite resolution (one can think of them as floating-point numbers with infinite word length). The computational trends suggest that the algorithm does not halt when using mathematical numbers, but this is a long way from a rigorous proof.

The algorithm *also works in three dimensions*. For that case one considers volumes *V* instead of areas *A* and follows the same steps given above. The first constructions of three-dimensional statistical geometry fractals were made by Paul Bourke, and can be seen at his web site [4].

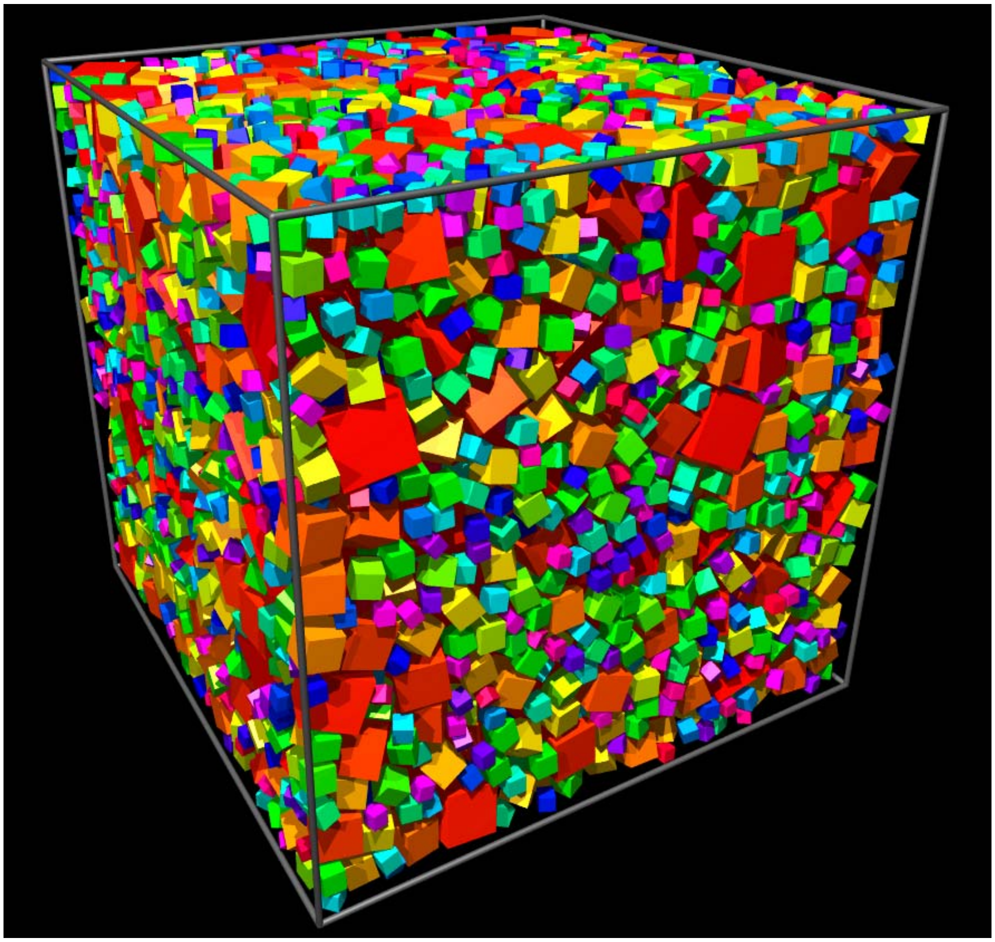
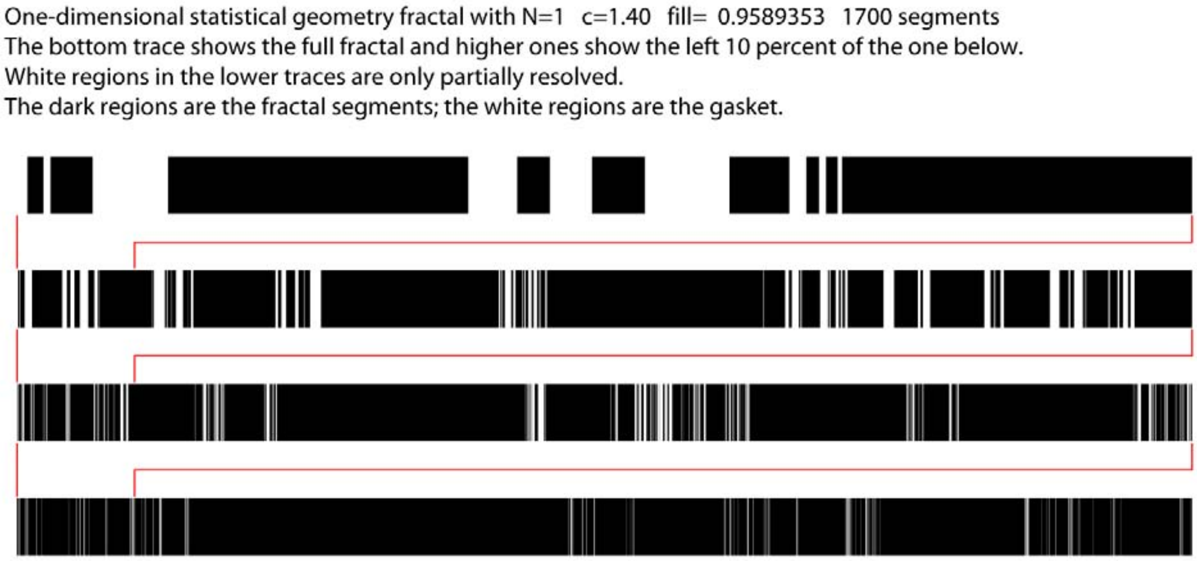


Fig. 3. A three-dimensional fractal where cubes with random orientation have been fractalized inside a cube. One can only see the cubes on the surface, which have been shaded for easier visualization. In three dimensions it requires placement of more shapes to achieve a given size range than in two dimensions. 5000 cubes, $c = 1.2$, $N = 2$. Courtesy of Paul Bourke.

The algorithm *also works in one dimension*. For that case one considers lengths *L* instead of areas *A* and follows the same steps given above. An illustration of a one-dimensional case is shown below. The result is a somewhat Cantor-like (but random) one-dimensional fractal.



One-dimensional statistical geometry fractal with N=1 c=1.40 fill= 0.9589353 1700 segments
The bottom trace shows the full fractal and higher ones show the left 10 percent of the one below.
White regions in the lower traces are only partially resolved.
The dark regions are the fractal segments; the white regions are the gasket.

Fig. 4. A one-dimensional statistical geometry fractal.

One The one-dimensional statistical geometry fractals are less interesting than the two-dimensional case as art, but from the viewpoint of theorem-and-proof mathematics they show most of the same features and offer a much simpler example.

It is found generally that the maximum usable *c* value varies strongly with dimension. For 1D, c values up to about 2.7 can be used. For 2D (squares) values up to around $c = 1.5$ are usable. For 3D (cubes) values can only run up to about $c = 1.2$.

2. Unique Features of the Statistical Geometry Algorithm

- By construction it is space-filling if it does not halt.
- Because of the power law it is fractal.
- It is random, not deterministic.
- It is not recursive.
- The shapes are non-touching (non-Apollonian).
- It allows use of a wide variety of shapes, and one can speak of "fractalizing" a shape or sequence of shapes.

The author is not aware of any other fractal which has the same properties. It is not a single fractal such as the Sierpinski gasket, but a large class of fractals with parameters *c* and *N* applicable to a wide variety of shapes. It may be the only non-Apollonian mathematically-defined fractal.

As far as the author can determine, fractals of this kind have not been described before his own discovery of them in 2010.

4. Boundaries.

In the rules it says to choose an area *A*. There are two ways to define the boundaries of this area. One way is to simply require all of the shapes to fall completely inside the boundary. Another way (for rectangular boundaries with dimensions *X* and *Y*) is to allow shapes to cross the boundary but insist that they be periodic, i.e. if a shape at *x*,*y* crosses the boundary other identical shapes must be included that are placed at $x \pm X$ or $y \pm Y$ or both. Using periodic boundaries one can tile the plane with copies of the rectangular base pattern (Fig. 6).

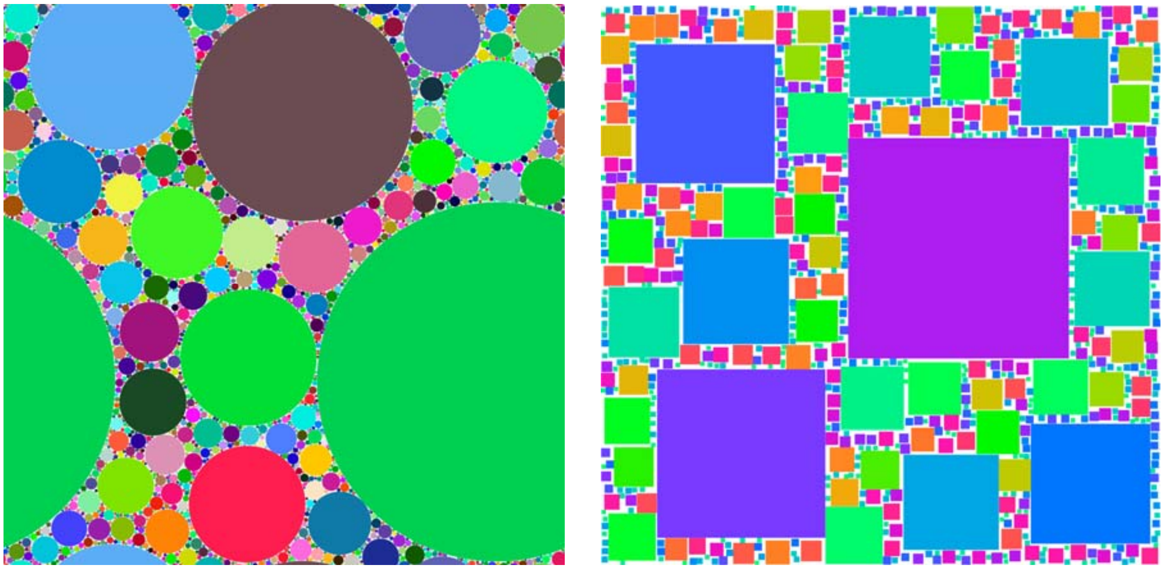


Fig. 5 On the left is a periodic boundary with circles, and on the right an inclusive boundary with squares. It can be seen that the largest (green) circle is in two pieces, and similarly for many of the smaller ones.

The algorithm works equally well with both kinds of boundary. For fractals with very high *c* values and inclusive boundaries the algorithm sometimes fails because of failure to find places for the first few large shapes. This is less often seen with periodic boundaries.

Inclusive boundaries are easier to program.

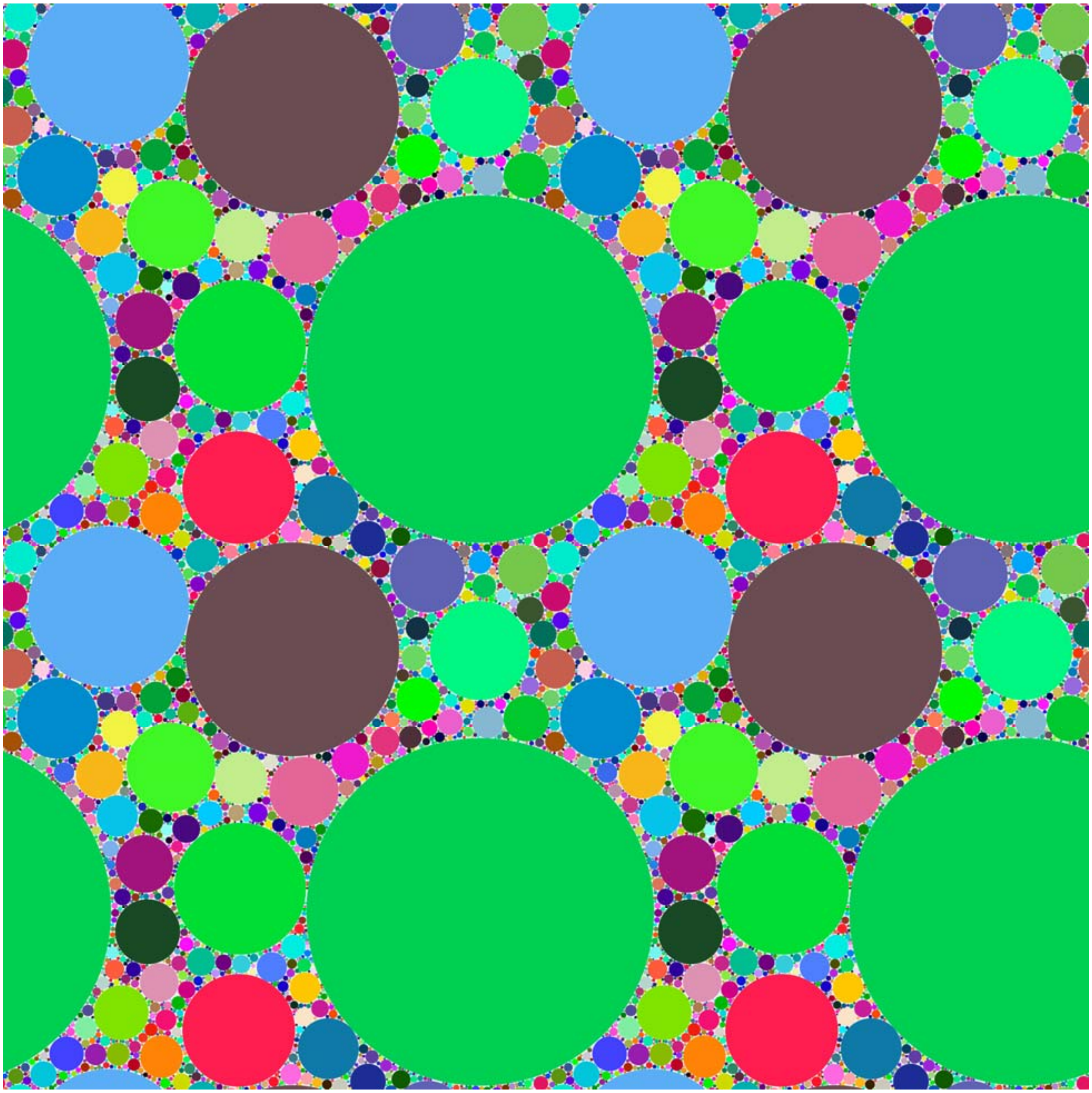


Fig. 6 Tiling of four of the periodic circles fractals of Fig. 5. Random color.

5. Color Schemes.

A central feature of these fractals is that there are shapes of all sizes, having the feature of "statistical self-similarity". How can you present the fractal image visually in such a way that the eye sees this?

One of the simplest approaches is to place black shapes on a white background (gasket). This works reasonably well if the number of shapes is modest, but if the fill factor exceeds ~90% the eye tends to blur the shapes together, and the smallest shapes are seen as gray.

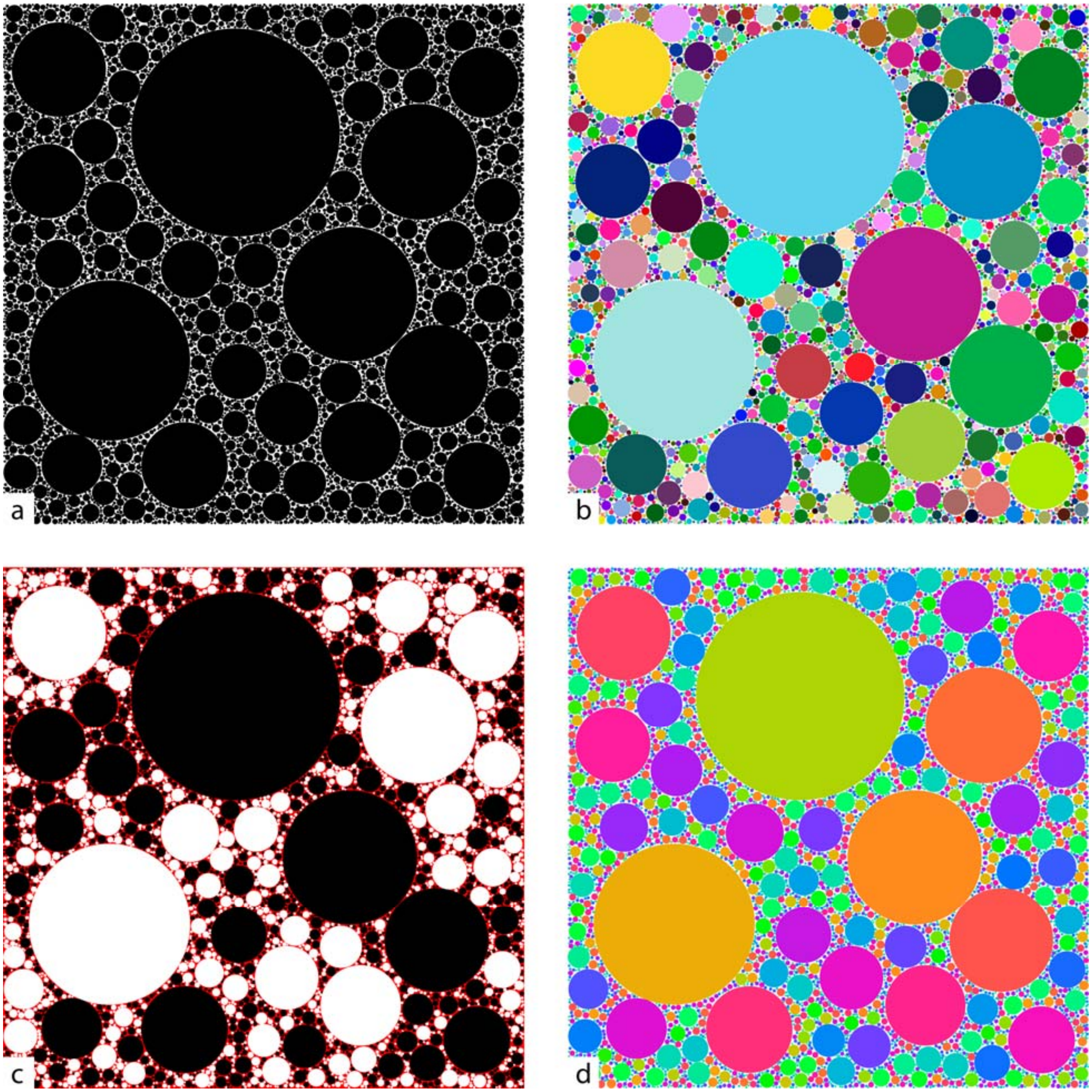


Fig. 7 Color schemes. (a) Black with white gasket; (b) Random colors with white gasket; (c) Black and white alternating, with a red gasket; (d) log-periodic such that circles about the same size are about the same color. These are all the same circle fractal.

A second method is to place alternating black and white shapes on a red background. This provides good visual contrast at all shape boundaries, but it does have the confusing feature that half the shapes have a different color although they are all part of the same sequence. It allows the eye to see that there are shapes of all sizes. This color scheme is quite useful when one has two shapes, such as mixed circles and squares, with one shape black and the other white.

A third color scheme is random color. The colors are chosen with equal probability for any location in RGB color space. This provides good contrast for shapes of all sizes. Black or white gasket.

A fourth approach is to use "log-periodic color". The power laws of a fractal become straight lines in logarithmic coordinates, and that is the basis for this color scheme. Suppose we deal with circles of radius r_i. One would then define a variable $u = log_{10}(r_i/r_0)$ and let the color be a periodic function of *u*. This causes all shapes of about the same size to have about the same color. The statistical self-similarity property is brought out by this color scheme, with each color having a similar distribution.

3D color schemes. This is a harder problem. The use of shading is quite helpful. Examples can be seen at Paul Bourke's web site [4].

6. Effect of the Parameter *c*.

The useful range of *c* values varies, depending on the shape. For hard-to-fractalize shapes the largest usable *c* value will be smaller. For art the useful range of *c* usually lies between about 1.15 and 1.4.

For large *c* the shape areas decrease more rapidly with *i*. The distance between shapes gets smaller as *c* increases, i.e., the packing is tighter [5], [7]. The closer shape-to-shape spacing and the smaller dimensionless gasket width mean that more trials are needed with high *c* for the placement of the *i*-th shape [9]. Correlation effects (see below) are more pronounced with high *c*.

It has been found that when one plots the cumulative number of trials n_{cum} needed to place *n* shapes versus *n*, for large *n* the data goes over to a power law with exponent *f* [9]. The value of *f* is always greater than *c*, and increases sharply as *c* increases. This is the basis for the claim that the algorithm never halts. For any number *n* of placed shapes, there is a predictable number *m* (sometimes very large) of trials needed to place them.

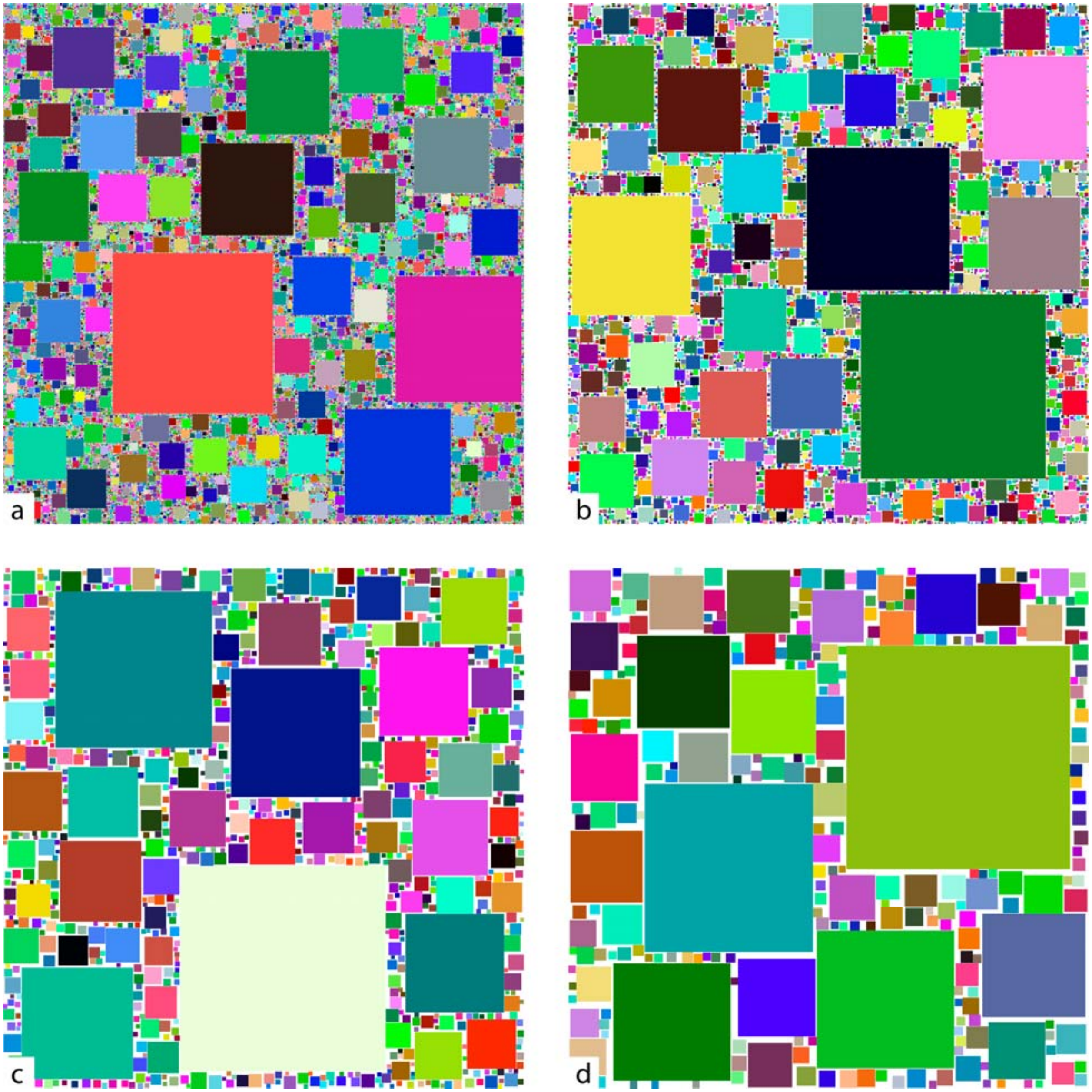


Fig. 8 Square fractals with different *c* values. Random color, white gasket. Fill = 88% in all cases. (a) $c = 1.20$, $N = 2$, 61518 shapes. (b) $c = 1.27$, $N = 2$, 3935 shapes. (c) $c = 1.34$, $N = 2$, 783 shapes. (d) $c = 1.41$, $N = 2$, 269 shapes. In (a) the smallest squares are not resolved.

7. Effect of the Parameter *N*.

The effect of increasing *N* is that the first shape is smaller and a larger number of placements is needed to achieve a given fill.

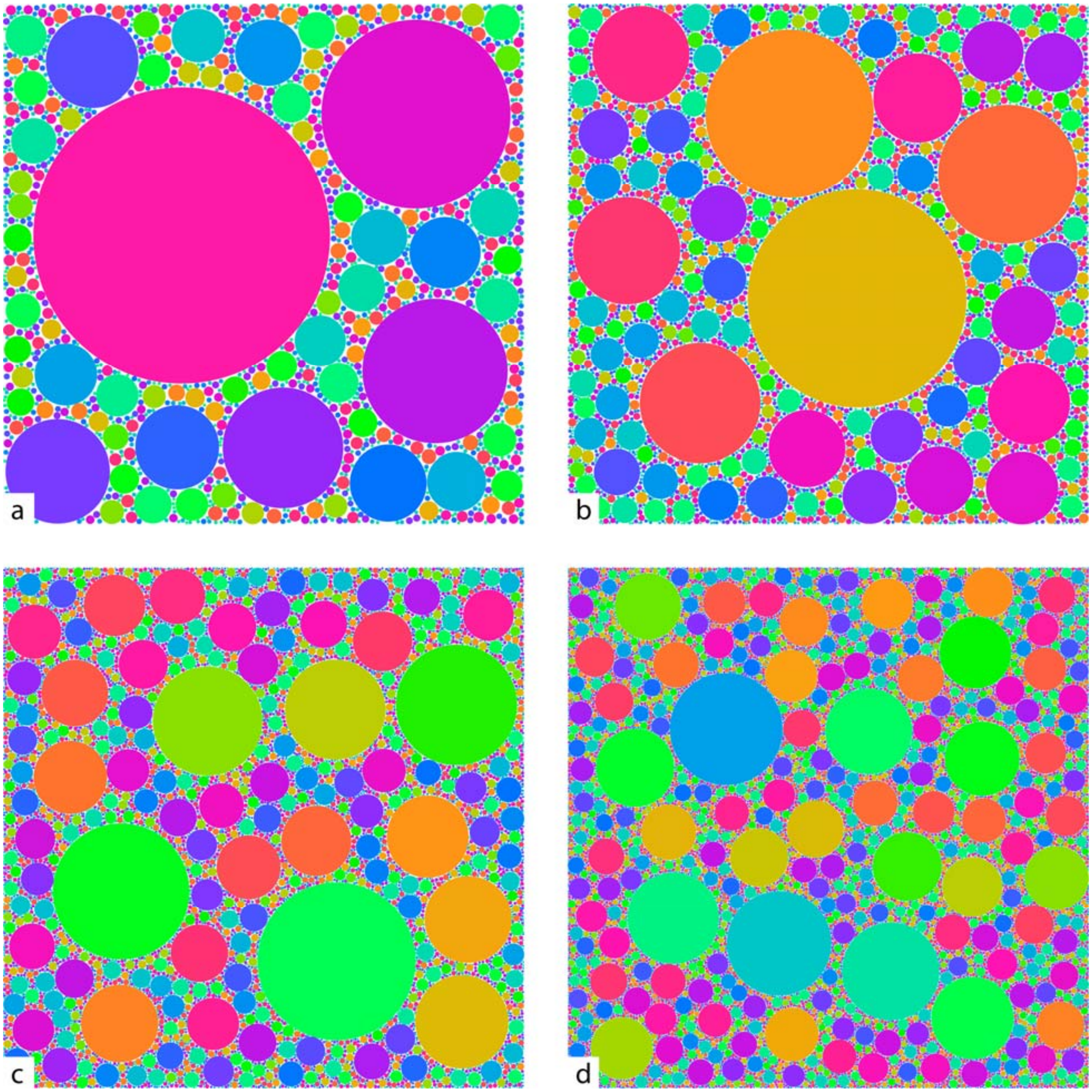


Fig. 9 Circle fractals with different *N* values. Log-periodic color. c = 1.3, fill = 90% in all cases. White gasket. (a) *N* = 1, 1224 shapes; (b) *N* = 2, 3297 shapes; (c) *N* = 4, 7580 shapes; (d) *N* = 8, 16265 shapes.

8. Self-Correlation. Rings.

There can be strong self-correlation for a single shape. This can be seen particularly well with rings. It was somewhat surprising (to me, anyway) that rings can be fractalized, but the algorithm ran smoothly. Here the correlation takes the form of "nesting" of smaller rings inside the bigger ones. It is quite obvious here that the nesting arises from the constraints of previous placements. Correlation is always stronger for higher *c* values.

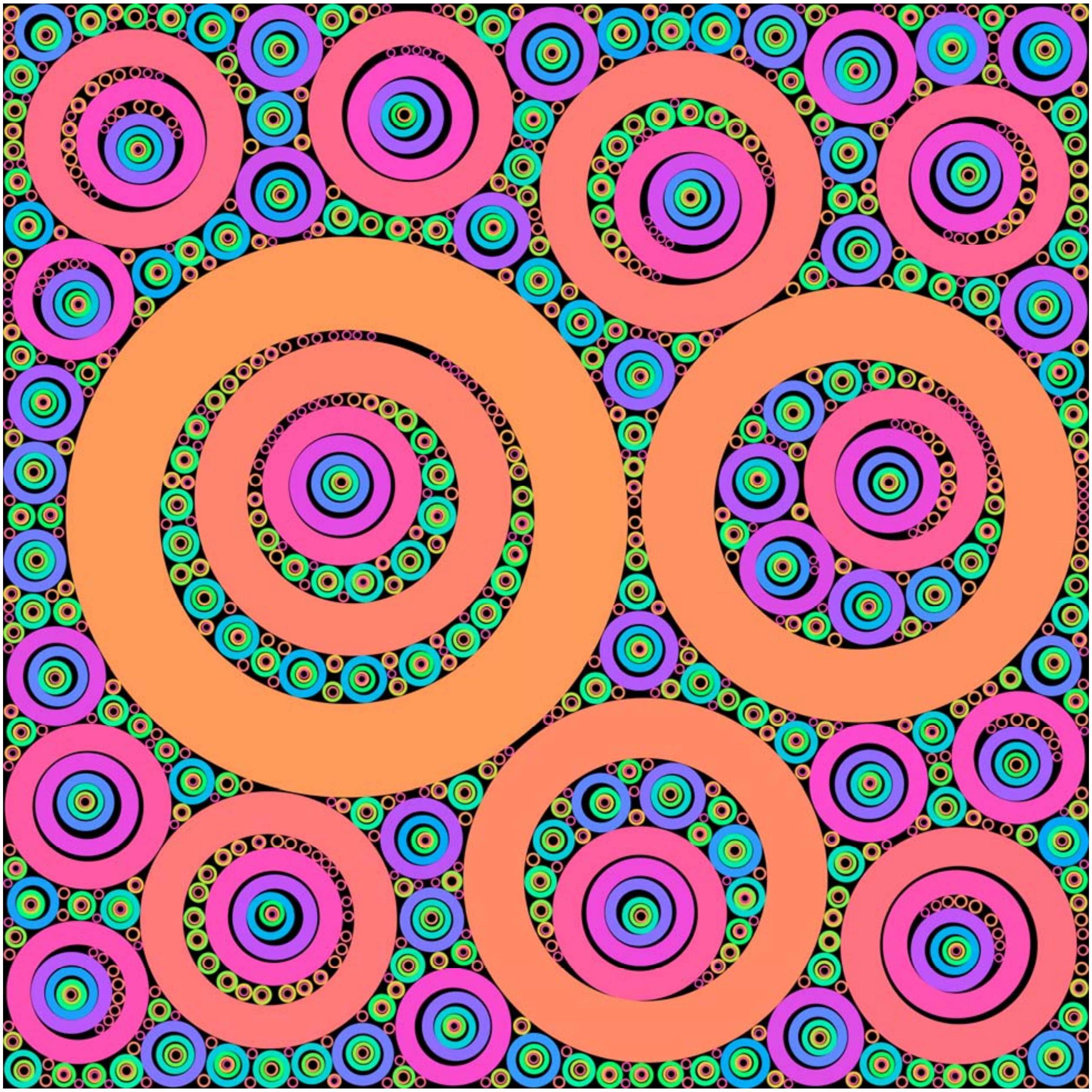


Fig. 10 Fractalized rings. $c = 1.25$, $N = 2$, 1500 shapes, fill = 82%. The inner radius is 2/3 of the outer radius. Log-periodic color. There is very strong self-correlation in the form of "nesting" of the rings. Black gasket, inclusive boundaries. A nice illustration of the constrained randomness of the algorithm.

9. Two Alternating Shapes Mixed. Mutual Correlation and Clustering.

An interesting discovery is that one can place two shapes alternating, as long as the area rule is satisfied. One might, for example, use circle-square-circle-square- … . Each shape would have the area specified for the *i*-th shape (see sec. 2).

Because the placement positions are constrained by all of the previous placements one finds a strong correlation in the positions of the two kinds of shape. For a mix of "up" and "down" triangles there is very strong anti-correlation. An "up" triangle will have hardly any "up" triangles for nearest neighbors[7] [7] but mostly "down" ones.

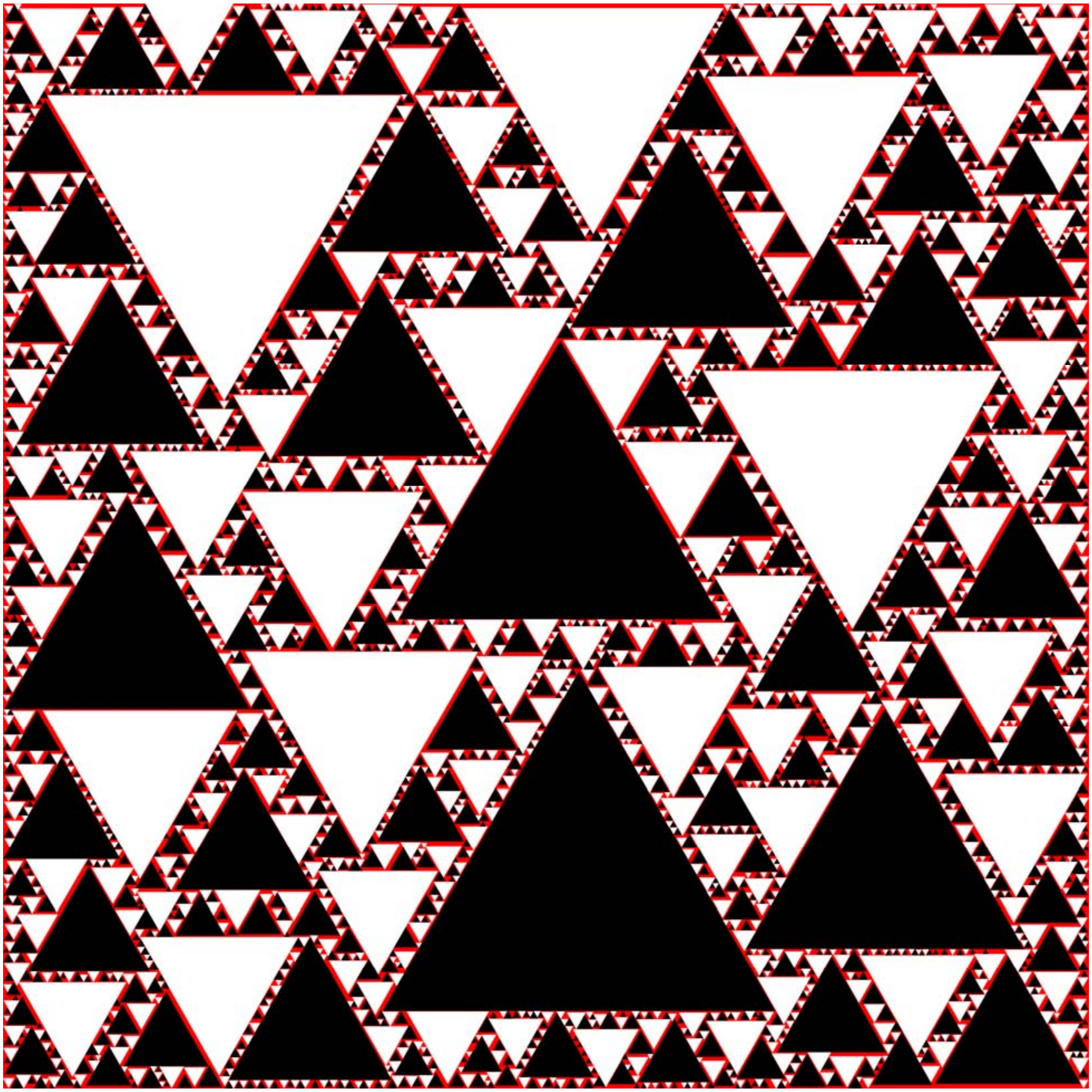


Fig. 11 Triangles with two orientations. The black triangles can be thought of as "up arrows" and the white ones as "down arrows". c = 1.35, N = 6, 3300 shapes, fill 89%. There is very strong anti-correlation of neighbors. Almost all the near neighbors of a black triangle will be white triangles and *vice versa*. Inclusive boundaries, red gasket.

[7] In crystallography or tessellations one can readily define nearest neighbors as those elements all lying at the same shortest distance from the center of a given element. It is evident that in these fractals a given shape can have many nearest neighbors (in the limit, an infinite number), and that none of them will be at the same distance as any other. A proper statistics-based definition of "nearest neighbor" remains to be found for these fractals.

For mixed circles and squares there is a positive correlation. Squares are placed most frequently near other squares, and the same for circles. Clustering is another description of the arrangement. Study of the results shows that more trials occur for circle placement than for square placement, indicating that squares are the more "packable" of these shapes. This is a *statistical* property of the square not contemplated by the Greeks.

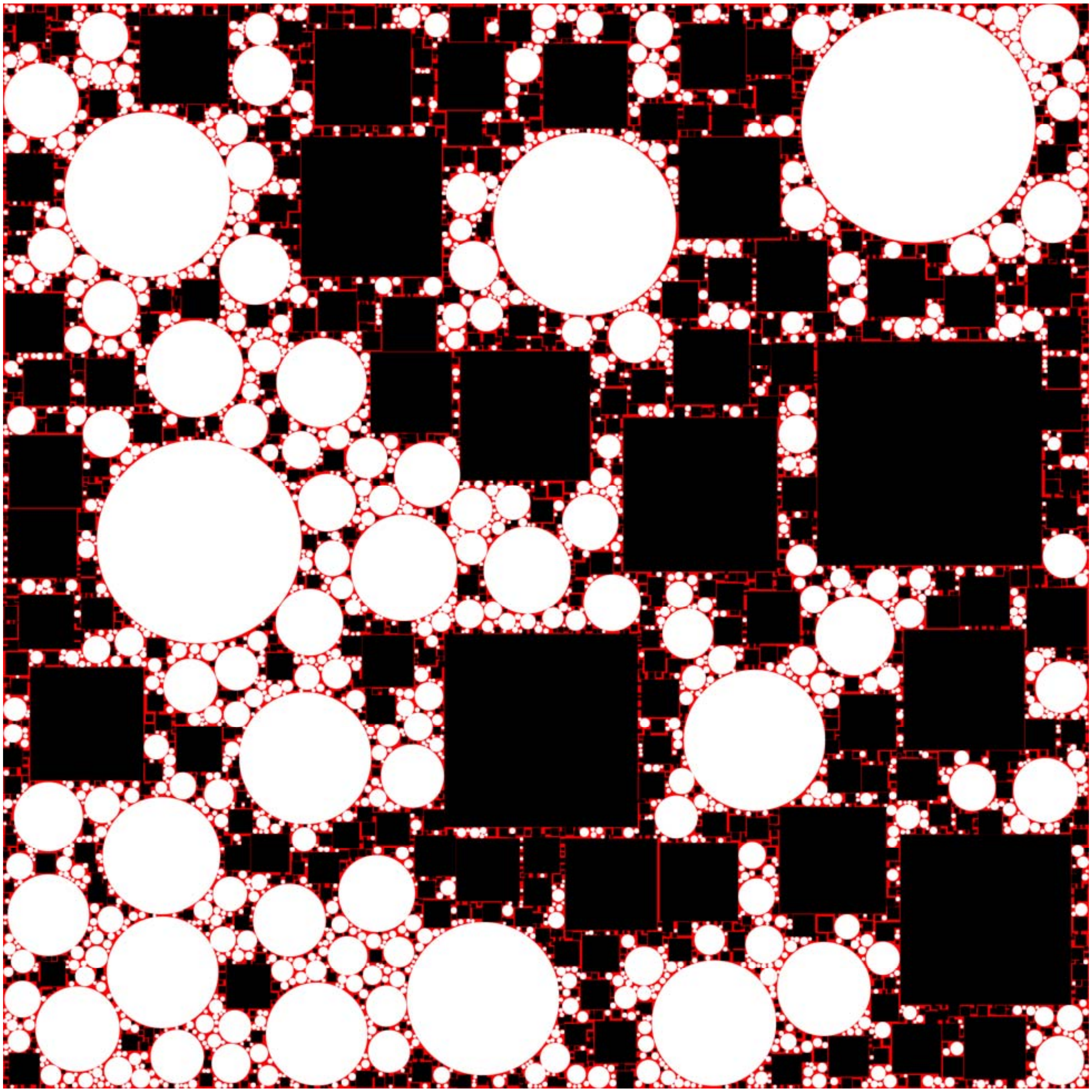
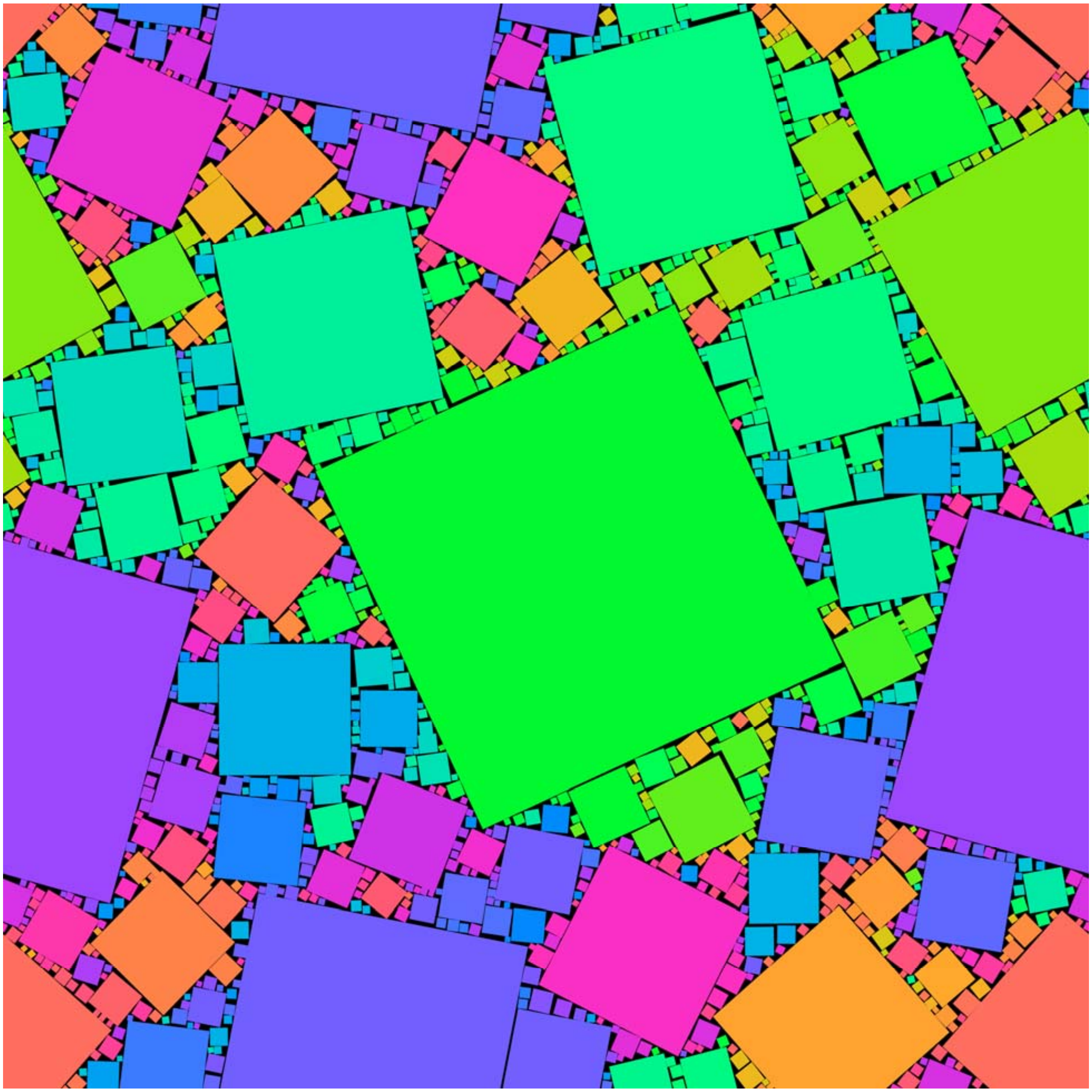


Fig. 12 Mixed circles (white) and squares (black). c = 1.35, N = 8, fill = 90%, 5425 shapes. Inclusive boundaries, red gasket. Note the strong mutual correlation, with circles mostly adjacent to circles and squares to squares. While the areas are "mostly white" or "mostly black", the minority shapes interpenetrate the others at all length scales. The degree of clustering depends strongly on the exponent c. The placement of the first few shapes strongly affects the overall pattern.

9. More Than Two Random Variables.

In the examples discussed above there are two random variables, x and y. It is possible to consider additional random variables. For example one can create a fractal with squares having different orientations[8], where the orientation angle is randomly chosen at each trial. Here there are three random variables. There is a clustering effect here. The first few large squares placed tend to set the orientation angle for the placements near them. The orientation effect is most pronounced for high c values.



[8] This was first studied by Paul Bourke.

Fig. 13. Squares with random orientation. c = 1.43, N = 3, 93% fill, 1500 squares. There is a special color scheme where the color is a continuous periodic function of rotation angle, so that squares of the same color have the same orientation. One sees "islands" of similar colors which show self-correlation. The first large squares to be placed strongly influence the orientations of later squares. Periodic boundaries, black gasket.

In another study [8] the shapes were defined in local polar coordinates with each shape defined by three phase angles so that no two shapes are the same. There are thus five random variables. The algorithm ran smoothly, leading to the conclusion that self-similarity or congruence of the shapes is not a requirement. One need only obey the area rule.

10. Run-Time Behavior.

What happens as the algorithm is executed? In particular, how many trials does it take to place some number of shapes? Does the algorithm halt? How is this behavior affected by c, N, shape, etc.?

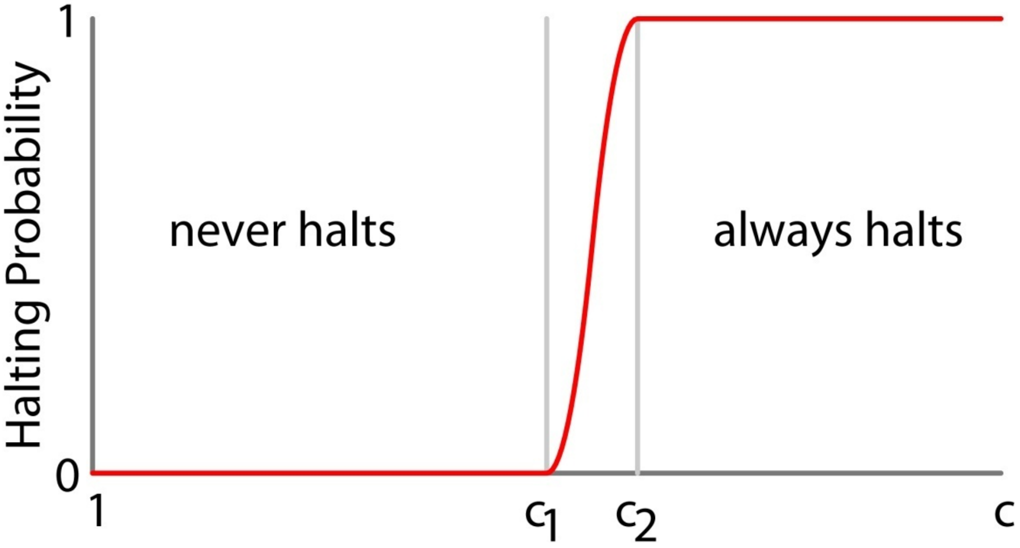


Fig. 13. Scheme for halting probability as a function of c. The values c_1 and c_2 will be high for simple shapes (circle, square) and low for sparse, sprawling shapes (e.g., quadcircle).

Halting has been found to obey the pattern shown in Fig. 13. This behavior has been studied in detail for the case of circles fractalized within a square as shown in Fig. 14. The most interesting feature is that there is a range of c values for which the algorithm never halts. This has been seen many times in the work on these fractals, but there is no deductive proof of it.

```
n_halt  c->  1   2   3    4    5    6    7    8     9    10    11
decade  1 -  0  10  51  109  207  343  511  783  1101  1480  1744
decade  2 -  0   0   0    0    0    1   16   57   125   163   150
decade  3 -  0   0   0    0    0    0    3    4    25    53    40
decade  4 -  0   0   0    0    0    0    0    1    14    22    21
decade  5 -  0   0   0    0    0    0    0    1     3    16    11
decade  6 -  0   0   0    0    0    0    0    0     1     5     1
             --- row 7 not printed ---
```

Total runs each c value: 2000

Maximum trials per placement = 8 000 000 Maximum placements = 70

c	1.28	1.30	1.32	1.34	1.36	1.38	1.40	1.42	1.44	1.46	1.48
halts	0	10	51	109	207	344	530	846	1269	1743	1969

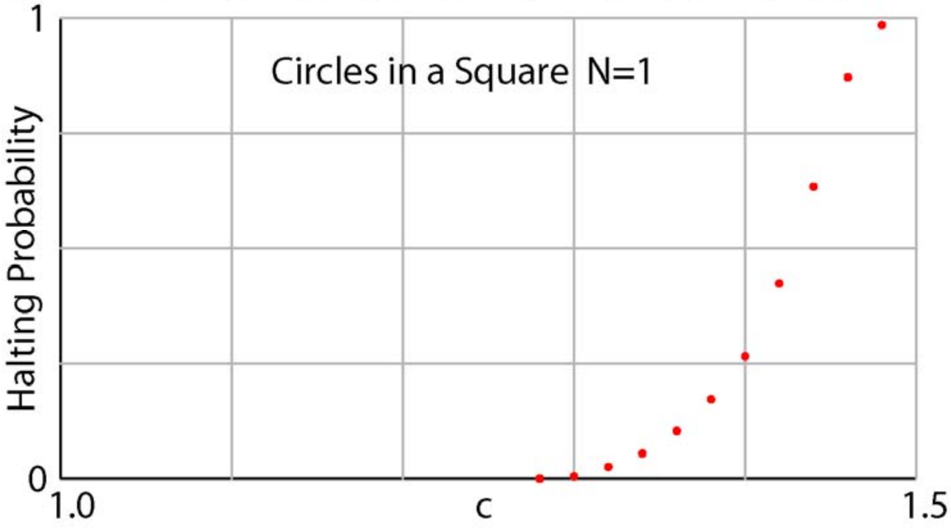


Fig. 14. Halting probability data. The points trace out a curve resembling Fig. 13.

The data of Fig. 14 has quite good statistical accuracy as 2000 runs were made at each *c* value. The raw data for the number of halting events is given in two rows above the graph. For example, there were no halting events for *c* = 1.28, and 1969 of them for *c* = 1.48. The numbers near the top give a breakdown of the number of placements at which the first halt event occurred. For example the column headed 9 refers to the 9th *c* value (*c* = 1.44) and we see that there were 1101 halting events with between 1 and 10 placements, 125 halting events between 11 and 20 placements, etc. This illustrates the observed fact that when halting occurs it almost always takes place during the early placements. If a run "survives" past 100 placements it will usually run on indefinitely.

We next turn to the question "For nonhalting runs, how many trials does it take to place *n* shapes?" Data on this is given in Fig. 15.

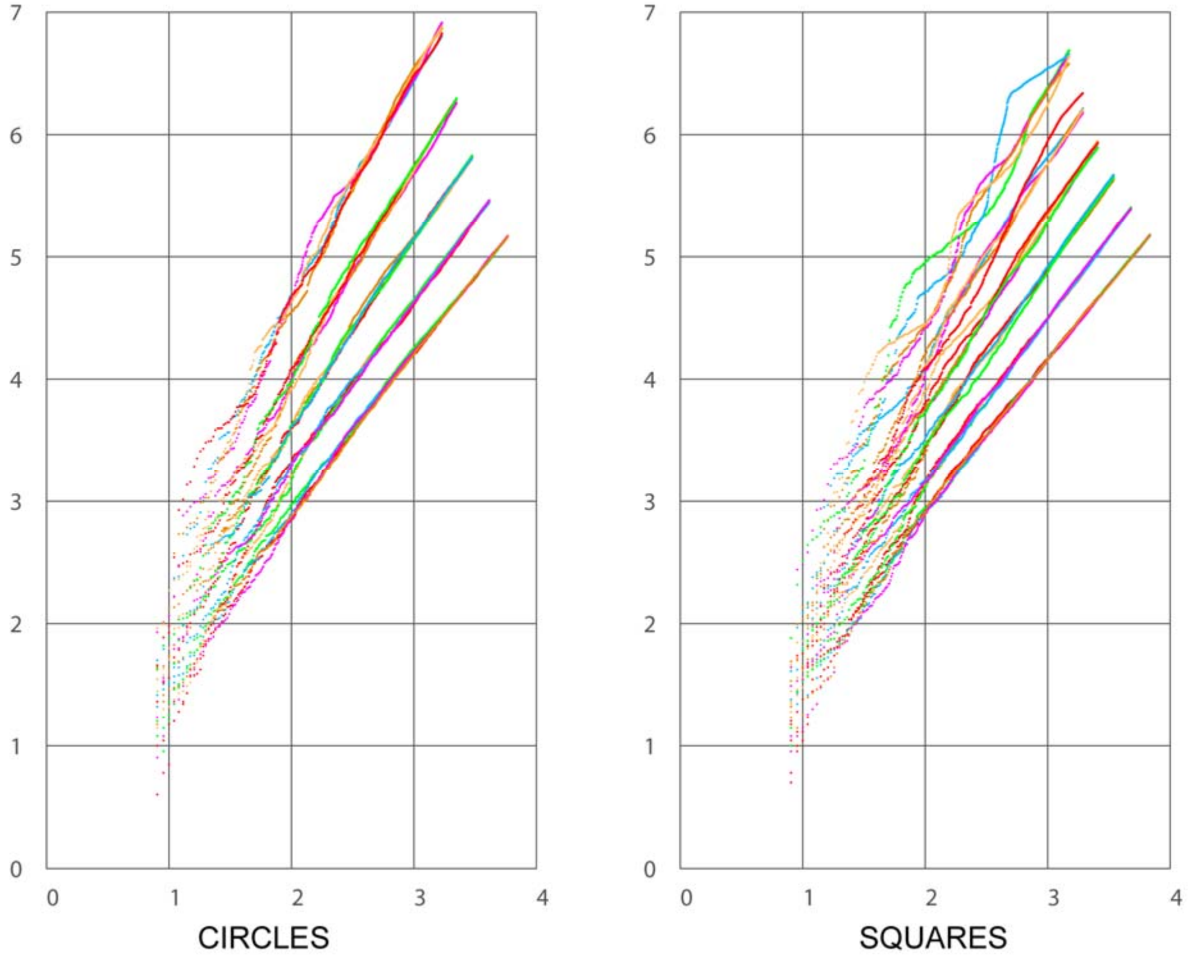


Fig. 15. Run-time records. $N = 1$ in all cases. The vertical coordinate is $log_{10}(n_{cum})$ where n_{cum} is the cumulative number of trials needed to place n shapes. The horizontal coordinate is $log_{10}(n)$. For the circles c = 1.20, 1.25, 1.30, 1.35, and 1.40 from bottom to top. Similarly for the squares c = 1.20, 1.25, 1.30, 1.35, 1.40, and 1.45. Five runs are plotted for each c value.

- For large n, the data follows a straight line, showing that $n_{cum}(n)$ follows a power law in n, i.e., $n_{cum}(n) = Kn^f$. The parameters K and f can be estimated from the data. They will have an uncertainty associated with the randomness of the process.

- For any number n of shapes to be placed one can calculate an expected number m of trials that will be needed. This is the basis for saying that the process does not halt. Although m may be huge, it is finite.

- It evidently takes fewer trials to place n squares than to place n circles when c is large. For the c = 1.2 data shape makes little difference.

- In all cases studied, $f \geq c$.

- The data becomes quite noisy for large *c*. It is thought that this noise sets an upper limit for usable *c* values.

This subject is discussed in more detail in [9], where relationships are found between *f* and *c* for circles and squares.

11. Fractal Dimension.

Based upon work with Apollonian[9] two-dimensional random fractals [10] it can be concluded that the fractal dimension *D* for two-dimensional statistical geometry fractals is

$$D = \frac{2}{c}$$

It also follows that *D* is independent of *N* and the same for any shape.

It is not obvious that the formulation of [10] is also valid for the non-Apollonian case, but I see nothing in the paper that precludes it. There is also data on the fractal *D* for circles found by box counting, supplied by Paul Bourke, which agrees with this equation within statistical uncertainty.

The fractal dimension formula $D = 1/c$ for the 1D case agrees well with box counting. If the equations of [10] are assumed to be valid for any number of physical dimensions, it then follows that $D = 3/c$ for the 3D case.

12. Invariance.

Invariance properties are always interesting. What properties of these fractals remain the same for any running of the algorithm, i.e. for any sequence of random numbers?

The first and most obvious item is that *the distribution of shape areas (or volume or length parameters) is invariant*. It is specified deterministically and will be the same for every run. The *fractal dimension* is also invariant. And the *dimensionless average gasket width b(c,N,n)* which for circles is

$$b(c,N,n) = \frac{(gasket\ area\ after\ n\ placements)}{(gasket\ perimeter\ after\ n\ placements)(radius\,of\ (n+1)^{th}\ circle)}$$

(discussed in more detail in [5]) is also invariant.

13. Conjectures and Problems.

Conjectures. All of these are supported by computational experiments, but lack any proof.

1. The algorithm runs "to infinity" without halting, with ideal mathematical numbers[10].

[9] The paper describes procedures for finding fractal *D* from computed sets of Apollonian fractal shapes. What I have used from this paper is equation (2). I have created data sets for the gasket area versus *n* and fitted it as they indicate to find β. Their equation (1) leads to the same result.

2. A power law is the sole, unique functional rule for area versus placed shape number *i* if the result is to be space-filling.

3. The algorithm works for any shape[11] or shape sequence if the area rule is followed.

Problems. Some of these may be beyond solution.

1. Find a mathematical scheme for describing the mutual correlation of the shape positions.

2. Define a numerical quantity which could be called the "packability" of a given shape, from which other properties could be calculated.

3. Find a way of computing parameters such as maximum *c*, exponent *f*, etc. from the shape alone.

4. Find a mathematical scheme for defining, identifying, and counting the nearest neighbors of a given shape.

In some cases the statements found here may be seen to conflict with the author's previous writings. This simply reflects closer study of the subject, which has caused some ideas to be scrapped and replaced by better ones.

14. References.

[1] "Fractals: Form, Chance and Dimension", Benoit Mandelbrot (1977). The *opus magnus* of fractals.

[2] John Shier, "Hyperseeing", Summer 2011 issue, pp. 131-140, published by ISAMA. A conference proceedings paper. Available by download at the web site john-art.com.

[3] "Statistical Geometry", John Shier, July 2011. A colorful self-published fractal art picture book available at lulu.com.

[4] Paul Bourke's fractal web site is **paulbourke.net**. The statistical geometry fractals are at **paulbourke.net/texture_colour/randomtile/**. Scroll to the bottom to see the 3D examples.

[5] "The Dimensionless Gasket Width *b(c,n)* in Statistical Geometry.", John Shier. Report 1 in a series. Available at the site john-art.com.

[10] Even a proof specialized to a single simple shape like the circle or square would be quite interesting.

[11] It would seem intuitively obvious that a sparse thin-branched shape such as a star with long, narrow points would be the worst-case shape. It is currently thought that each shape has a given maximum *c* value, which is lower for many-branched shapes. The view is taken that any value $c > 1$ (i.e., shapes for which the zeta function is convergent) is a "legal" value of *c*. If one examines fractals with low *c* values one sees that they are quite sparse and forgiving of branched shapes. This is the basis for the claim that any shape can be fractalized. This may be overstated, but no existing computational evidence refutes it. Proof of such a conjecture would appear to be an exceedingly difficult undertaking.

[6] "Holes in Circle Fractals. Statistics.", John Shier. Report 2 in a series. Available at the site john-art.com.

[7] "Nearest Neighbor Distances in Circle Fractals. Statistics.", John Shier. Report 3 in a series. Available at the site john-art.com.

[8] "Multishapes and Packability in Statistical Geometry.", John Shier. Report 4 in a series. Available at the site john-art.com.

[9] "Statistics of Trials and Placements: Circles and Squares. The Parameters *c* and *f*.", John Shier. Report 5 in a series. Available at the site john-art.com.

[10] "Relation Between Grain Shape and Fractal Properties in Random Apollonian Packing with Grain Rotation", Gary W. Delaney et al., Phys. Rev. Letters, **101**, 120602 (2008)

John Shier

September 2011 (Rev 5 January 2013)