

Statistical Geometry in One Dimension

An algorithm which produces a family of Cantor-like one-dimensional fractals on a line segment is described. Random numbers are a key feature.

One-dimensional statistical geometry fractal with $N=1$ $c=1.60$ $\text{fill}= 0.98740$ 864 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. 1. An example of the fractal with $c = 1.6$. Several scales of detail are shown so that the reader can better grasp its form.

1. 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]. An understanding of the material presented requires a grasp of the rules for constructing the fractals:

1. Create a sequence of line segment lengths λ_i following a power law, equal to

$\frac{1}{N^c}, \frac{1}{(N+1)^c}, \frac{1}{(N+2)^c}, \frac{1}{(N+3)^c}, \dots$ where c and N are constant parameters. A line of length L is to be filled with such segments.

2. Sum the lengths λ_i to infinity, using the Hurwitz zeta¹ function

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.

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

3. Define new lengths L_i by $L_i = \frac{L}{\zeta(c, N)(N+i)^c}$. It will be seen that the sum of all these redefined lengths is L .

4. Let $i = 0$. Place a segment having length L_0 in the length L at a random position x such that it falls entirely within length L . This is the "initial placement". Increment i .

5. Place a segment having length L_i entirely within L at a random position x such that it falls entirely within L . If this segment overlaps with any previously-placed segment repeat step 5. This is a "trial".

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

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

This is a very *simple* algorithm, easily stated in a few lines of text. Some might think it should be stated in terms of sets, but since what is actually reported are the results of computational experiments the results are described in those terms. All of the computations have been done with double-precision arithmetic.

The parameters c and N can have a variety of values. In practice the parameter c is usually in the range² 1.2-2.0. N can be 1 or larger, and need not be an integer.

By construction the result is a space-filling random fractal -- *if the process never halts*³. Available evidence (see below) says that it does not halt, at least for c values which are not close to the upper limit of usable c values.

The one-dimensional case probably provides the simplest structure for proofs of the various properties of the algorithm.

2. Unique Features of the Algorithm

- By construction it is space-filling if it does not stop.
- Because of the power law it is fractal.

² If $c < 1.2$ the number of trials needed to achieve a substantial fill becomes huge. If $c > 2.0$ the number of trials per placement becomes huge.

³ There are two ways to consider the halting question. One can ask whether the *computational algorithm* using floating-point numbers stops. 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 stops for ideal mathematical 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 stop when using ideal numbers, but this is a long way from a rigorous proof.

- It is random, not deterministic.
- It is not recursive.
- The segments are non-touching.

The author is not aware of any other fractal which has the same properties. It is not a single fractal such as the Cantor fractal, but a large class of fractals with parameters c and N applicable. The algorithm also works in two and three dimensions [2][3][4] where it produces non-Apollonian fractals.

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

3. Boundaries.

In the rules it says to choose a length L . There are two ways to define the boundaries. One way (inclusive boundaries) is to simply require all of the segments to fall completely inside the end points. Another way (periodic boundaries) is to allow segments to cross the boundaries but insist that they be periodic, i.e. if a segment at x crosses the boundary another identical segment must be included that is placed at $x \pm L$. The work reported here uses inclusive boundaries.

4. Effect of the Parameter c .

For large c the segment lengths decrease more rapidly with i . The average distance between segments gets smaller as c increases, i.e., the packing is tighter. The closer segment-to-segment spacing and the smaller dimensionless gasket [5] width mean that more trials are needed with high c for the placement of the i -th segment.

The one-dimensional algorithm can produce very high fill factors (up to 99.999%) compared to 2D and 3D.

5. Statistical Distribution of the Gaps.

By gaps we mean the regions which make up the gasket. With inclusive boundaries the parts of the gasket at the ends of L are also viewed as gaps.

Before the initial placement there is just one gap of length L . The initial placement replaces this with two gaps. Each subsequent placement eliminates one gap and replaces it by two smaller ones. Thus after n placements the number of gaps will be $(n + 1)$.

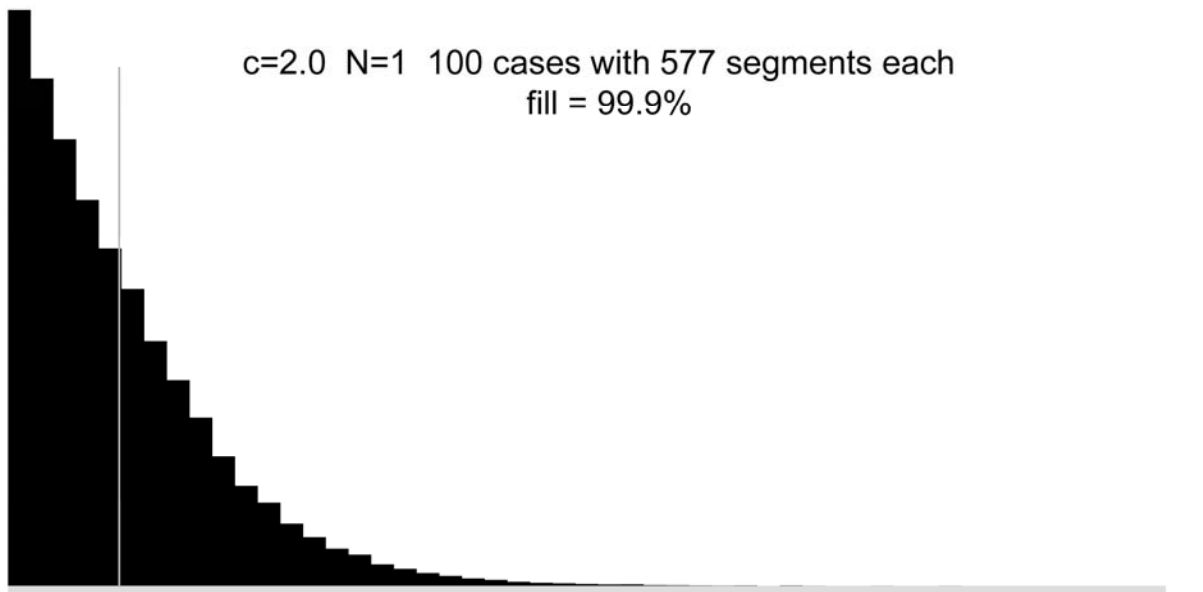


Fig. 2. A histogram of the gap widths. The vertical coordinate is the number of instances. The gray line underneath shows the extent of the data. The largest gap lies in the farthest-right bin. The vertical gray line shows the width of the last-placed segment on the same x scale.

It can be seen that the data follows a decaying exponential trend. There are many gaps larger than the next-to-be-placed segment (vertical gray line), indicating that the algorithm will have no difficulty placing more segments. This is typical of the histograms for any c and any number of placed segments.

There is a sparse population of quite large gaps, which are improbably numerous if this is truly an exponential p.d.f. There is a largest gap at any stage of the algorithm.

6. Run-Time Behavior.

What happens as the algorithm is executed? In particular, how many trials does it take to place some number of segments? How is this behavior affected by c ? To this end one can plot $\log_{10}(n_{cum})$ versus $\log_{10}(n)$ where $n_{cum}(n)$ is the total number of trials needed to place n segments. Such a record will be different for each run.

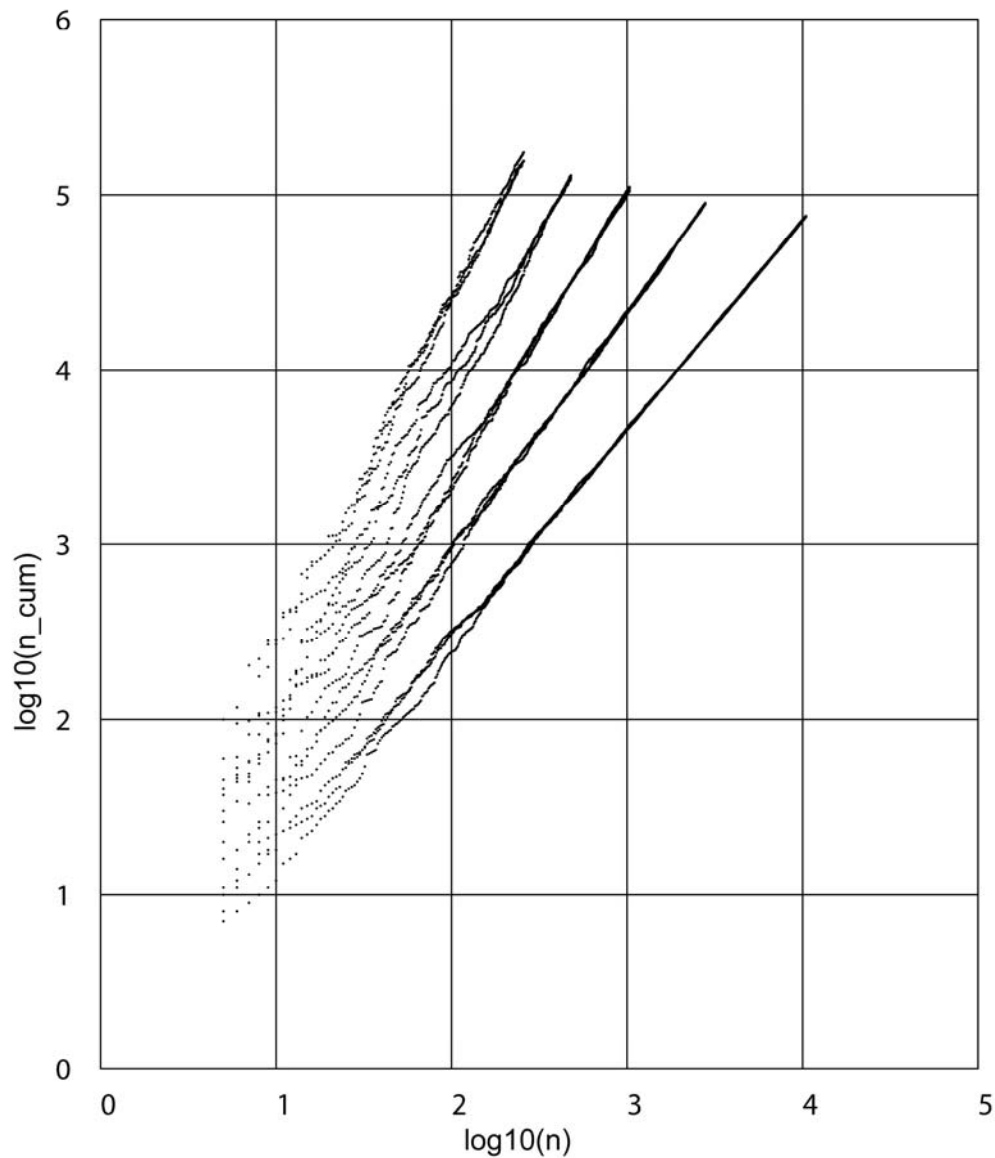


Fig. 3. Run-time records. $N = 1$ in all cases. The vertical coordinate is $\log_{10}(n_{cum})$ where n_{cum} is the cumulative number of trials needed to place n segments. The horizontal coordinate is $\log_{10}(n)$. From bottom to top, $c = 1.2, 1.4, 1.6, 1.8,$ and 2.0 . Three runs are plotted for 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 segments 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.
- For all of the c values on the graph, $f \cong c$ within statistical error.

- Lines fitted to the data appear to pass through the origin of (logarithmic) coordinates (0,0).
- The data becomes quite noisy for large c . It is thought that this noise sets the upper limit for usable c values.

7. Fractal Dimension.

I have received box counting estimates of the fractal dimension D for the one-dimensional case from Paul Bourke [4]. This data fits reasonably well (within statistical uncertainty) to the formula $D = 1/c$. The largest c values appear to be around 2.7, which translates to a fractal D of .37.

8. Conjectures and Problems.

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

1. The algorithm runs "to infinity" without stopping, with ideal mathematical numbers.
2. A power law is the sole, unique functional rule for segment length versus placed segment number if the result is to be space-filling.

Problems.

1. Show that the mean or most probable value of f is c when $N = 1$.

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.

9. 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. 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.