Statistical Geometry Fractals in Three Dimensions

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 volumes υ_i following a power law, equal to $\frac{1}{N^c}, \frac{1}{(N+1)^c}, \frac{1}{(N+2)^c}, \frac{1}{(N+3)^c}, \ldots$ where c and N are constant parameters. A volume V is to be filled with such volumes.

2. Sum the volumes υ_i to infinity, using the Hurwitz zeta[1] function

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

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

4. Let $i = 0$. Place a shape having volume V_0 in the volume V at a random position x,y,z such that it falls entirely within volume V. This is the "initial placement". Increment i.

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

6. If this shape <u>does not</u> overlap with any previously-placed shape, store x,y,z and the linear dimensions of the shape in the "placed shapes" 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.

The parameters c and N can have a variety of values. In practice the parameter c is usually in the range 1.1-1.15 with three spatial dimensions. 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*[2]. 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.

---

[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. Results and Discussion.

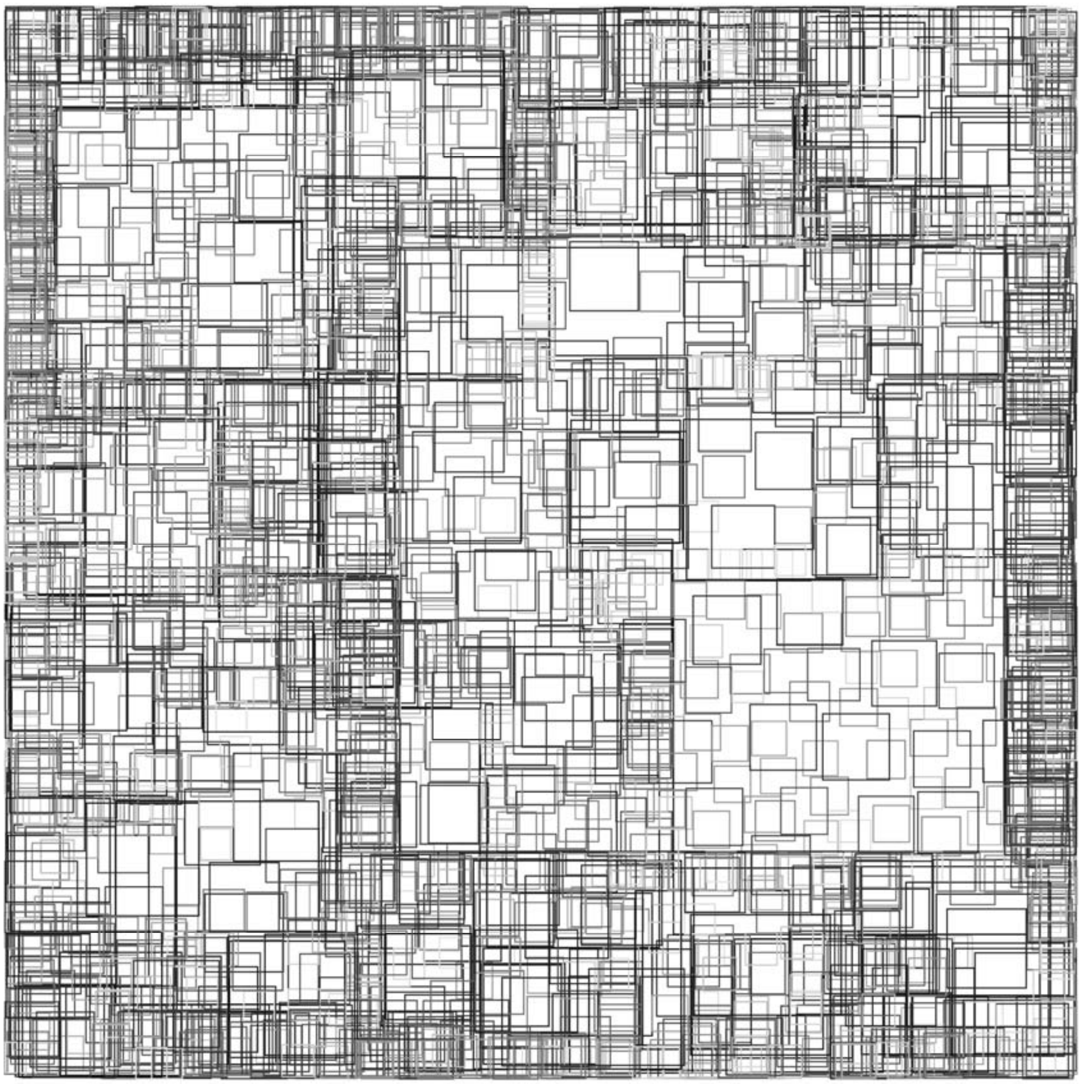

Fig. 1. Image of fractal cubes packed within a cube. This is the simplest and fastest-running 3D statistical geometry fractal. $c = 1.155$, $N = 3$, 2394 cubes, fill 78.3%. This is the "x-ray" view down the z axis, with nearer objects having a darker color. There are fewer features in the areas near large cubes because there is less room there.

[2] 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.

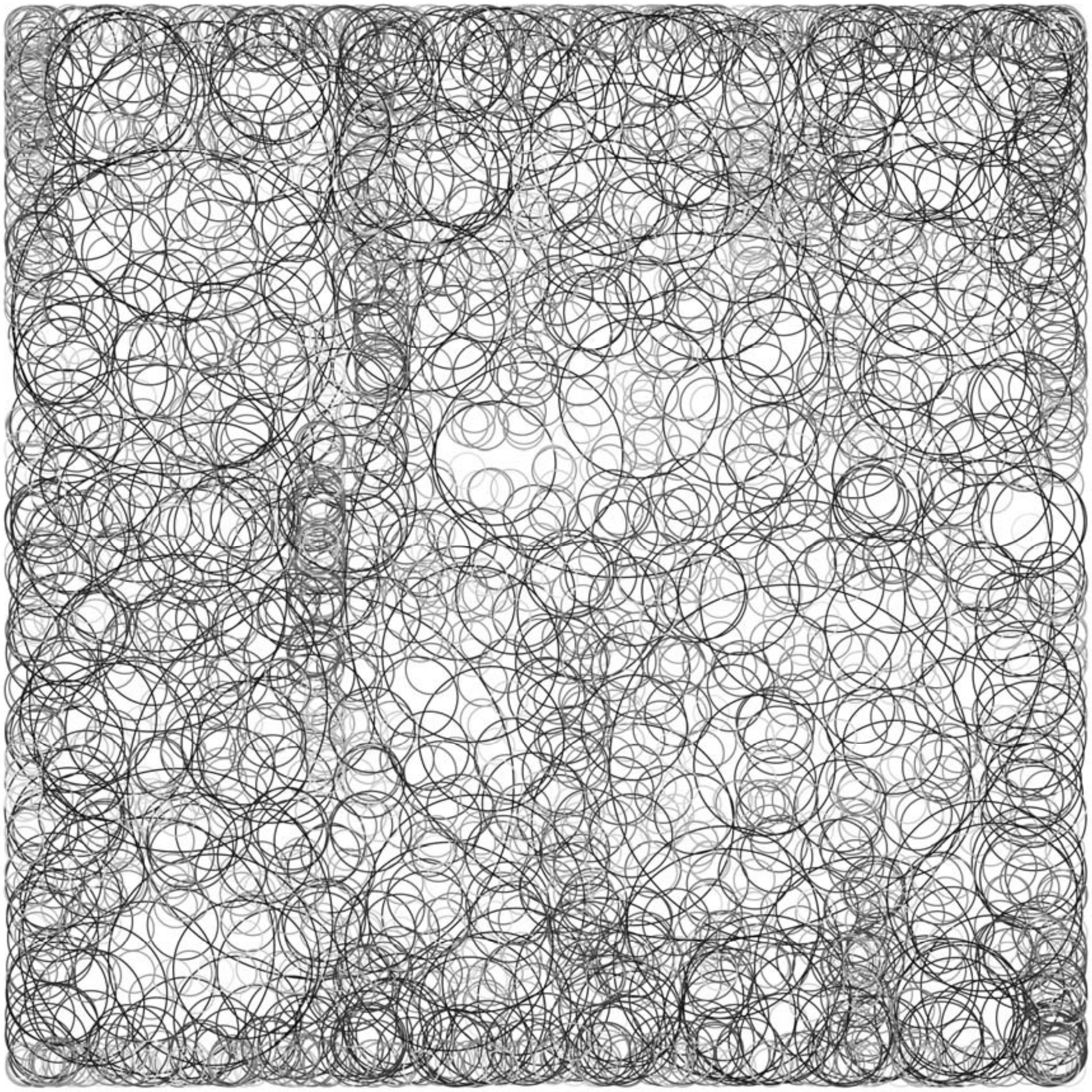

Fig. 2. Image of fractal spheres packed within a cube. $c = 1.13$, $N = 3$, 2844 spheres, fill 70.3%. This is the view down the z axis, with nearer objects having a darker color. There are fewer features in the areas near large spheres because there is less room there.

The algorithm ran quite smoothly for both cubes and spheres packed into a cube. For cubes with inclusive boundaries the largest usable c value is about 1.155, while for spheres it is about 1.13. With periodic boundaries and $N = 1$ the largest usable c value for cubes was found to be about 1.207. This is substantially lower than for the 2D case. In 3D cubes have better "packability" than spheres, which is similar to squares and circles in the 2D case.

In general it is harder to get large size ratios of largest/smallest shape and large fill factors compared to the 2D and 1D cases.

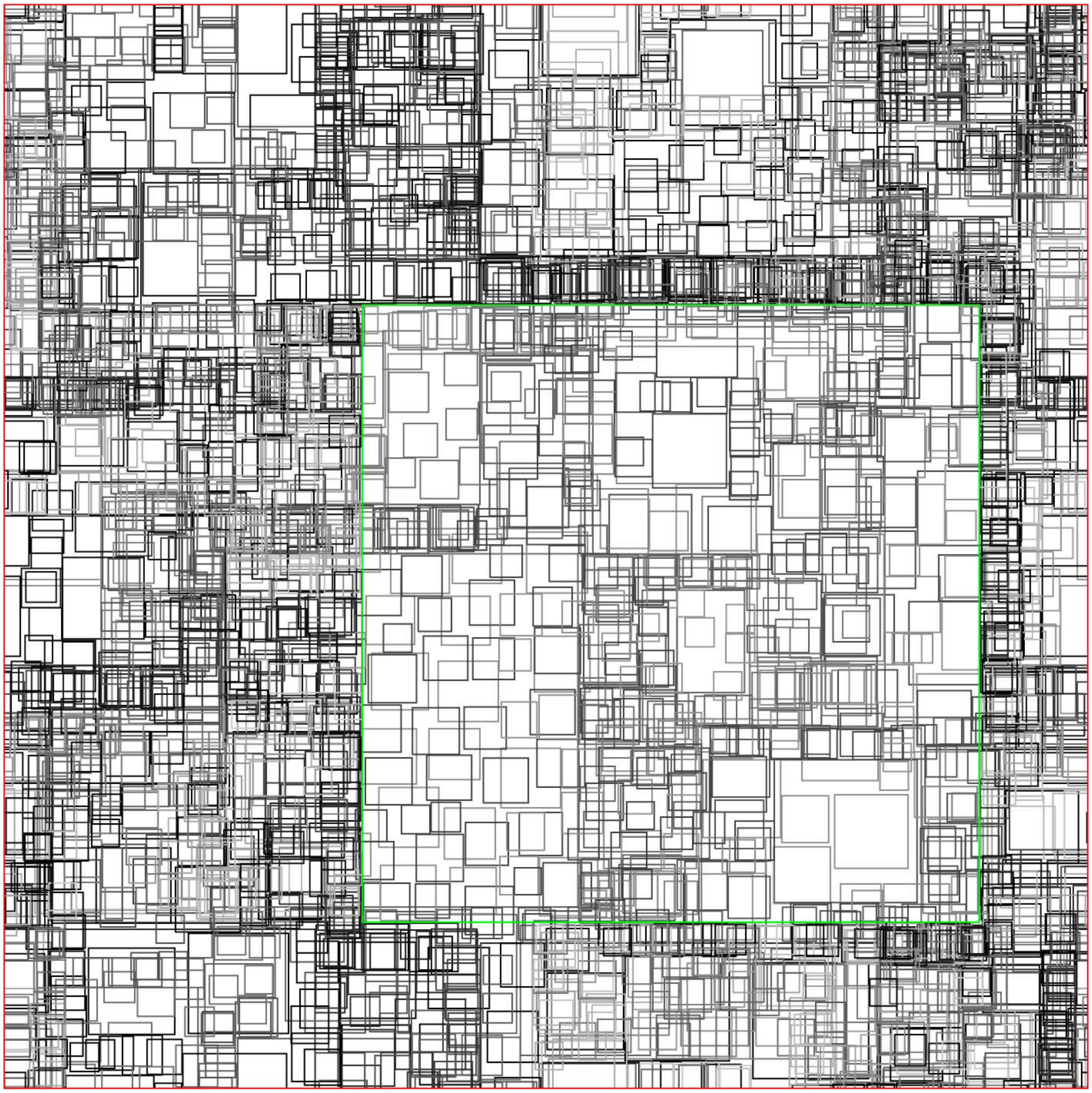

Fig. 3. Image of fractal cubes packed within a cube with periodic boundaries. *c* = 1.207, *N* = 1, 2500 cubes, fill 82.3%. This is the view down the *z* axis, with nearer objects having a darker color. There are fewer features in the areas near large cubes because there is less room there. The red line is the cell boundary, while the green line shows the first, largest cube.

If we assume fractal *D* to be given by $D = 3/c$, we find (with inclusive boundaries) that the smallest *D* is 2.61 for cube packing, and 2.65 for sphere packing. It is noted that Apollonian packing of spheres has a fractal *D* of 2.47, and since Apollonian packing can be viewed as fractal "closest" packing it is plausible to assume that for the sphere case $D > 2.47$, i.e., it sets a largest usable *c* value ($c < 1.21$) for non-Apollonian random sphere packing.

More elegant three-dimensional statistical geometry fractal images with shading, oblique viewpoints, etc. can be found at Paul Bourke's web site [4]. Several of the images show cases where cubes or tetrahedra have been given arbitrary rotation angles.

3. 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.

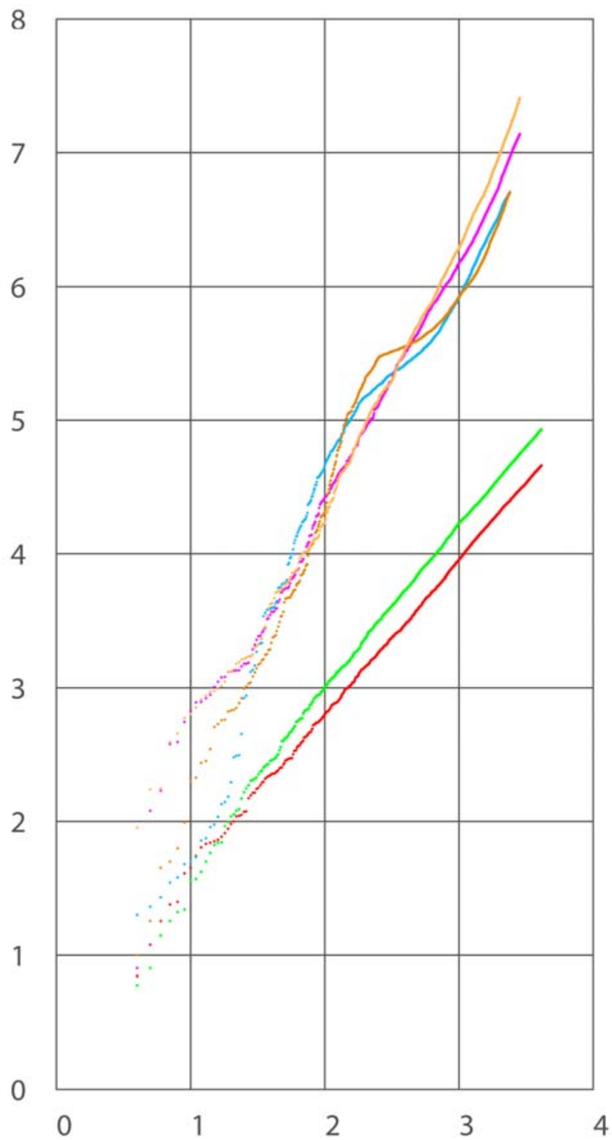

Fig. 3. Run-time records. N = 3 in all cases. Inclusive boundaries. The vertical coordinate is $log_{10}(n_{cum})$ where n_{cum} is the cumulative number of trials needed to place *n* segments. The horizontal coordinate is $log_{10}(n)$. The red and green records are for c = 1.08. The red one is for cube placement and the green one is for sphere placement. The upper records are for spheres with c = 1.13 and cubes with c = 1.155 (these values are close to the respective limits of usable *c* values).

- For large *n*, the data follows an approximate 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 c = 1.08, $f \approx c$ within statistical error. For the higher *c* values *f* is around 2.
- The data becomes quite noisy for large *c*. It is thought that this noise sets the upper limit for usable *c* values.

4. Fractal Dimension.

The author believes, based upon interpretation of certain published papers, that $D = 3/c$. This needs confirmation.

5. References.

[1] "The Fractal Geometry of Nature", 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.