PHYSICAL REVIEW A VOLUME 45, NUMBER 10 15 MAY 1992

ARTICLES

Box-counting multifractal analysis

L. V. Meisel, Mark Johnson, and P. J. Cote
Research, Development and Engineering Center, Close Combat Armament Center, Benet Laboratories, Watervliet, New York 12189
(Received 13 May 1991)

Two box-counting algorithms for the determination of generalized fractal dimensions are described. Results of application of the algorithms to Euclidean curves, quadric islands, Koch symmetric and asymmetric triadic snowflakes, and split snowflake halls introduced by Mandelbrot [*Fractal Geometry of Nature* (Freeman, New York, 1983)] are described. Comparison to analytic results for the model curves is provided and the effectiveness of the algorithms is discussed.

PACS number(s): 02.70.+d, 05.45.+b

INTRODUCTION

The accomplishments of fractal modeling in materials to date are impressive. However, characterization of fractal structure in terms of the "fractal dimension" is incomplete. The "fractal dimension" tells us how the length, area, or density of an object varies with scale, but much more information is required to characterize or model the more complicated structures that occur in nature. The theory of multifractal objects provides a more comprehensive description of fractal objects. The major efforts in multifractal analysis have been directed toward the understanding of the subset of the more complicated fractal objects described in Halsey, Jensen, Kadanoff, Procaccia, and Shraiman [1].

In order to optimally exploit multifractal perspectives, it is imperative that quantitative techniques for the measurements of fractal properties be developed. Box-counting techniques for the measurement of generalized fractal dimensions render the most direct application of the ideas in Halsey *et al.* [1].

Block, von Bloh, and Schellnhuber [2] (BBS) have described a box-counting algorithm for the measurement of generalized (fractal) dimensions, which they refer to as "efficient box counting" (EBC). The present work describes two other realizations of box-counting procedures and addresses techniques for evaluating generalized fractal dimensions over wider ranges of the controlling parameters than that covered in the work of BBS. The multifractal models employed to test the effectiveness of the algorithms are subject to simple analysis along the lines in Halsey *et al.* [1], so that our "measurements" are compared directly with analytic results.

THEORY

Fractal measures

Although one can conceive of a more general class of fractal point sets than that analyzed in the seminal work of Halsey, Jensen, Kadanoff, Procaccia, and Shraiman [1], it provides a basis for the analysis of most of the known fractal point sets, curves, etc. In particular, it provides the theoretical basis for multifractal analysis of the Koch constructions studied here. Following Halsey *et al.* [1], we define a partition function

$$\Gamma(q,\gamma,L)=\sum_i \frac{p_i^q}{L_i^\gamma} \quad \text{where } \gamma=(1-q)D_B(q) \ . \tag{1}$$

Then, for recursive constructs,

$$\Gamma(q,\gamma,L^n)=[\Gamma(q,\gamma,L)]^n \quad \text{so that } \Gamma(q,\gamma(q),L)=1 \ . \tag{2}$$

The generators of the multifractal models studied here have identical weights p_i, which sum to unity, and have two length scales. Letting n_i be the number of elements in the generator characterized by the length scale L_i, the intrinsic values of $D(q)$ are obtained by solving the transcendental equation

$$p^q[n_1L_1^{-\gamma}+n_2L_2^{-\gamma}]=1 \tag{3}$$

for $\tau(q)=(q-1)D_B(q)$. This is a trivial problem, however, one could avoid dealing with transcendental equations and solve

$$q(\gamma)=-\log_{10}[n_1L_1^{-\gamma}+n_2L_2^{-\gamma}]/\log_{10}[p] \tag{3a}$$

for $q(\gamma)$. For multifractals having many scales and weights, solution of a transcendental equation is required.

Box counting

Box-counting methods for the determination of generalized fractal dimensions are, in principle, well known. Recently, Tobochnik and Gould [3] presented a valuable pedagogical review of the underlying concepts of box-counting methods such as those employed in the present study. The details of the numerical procedures employed here are provided below.

Numerical methods

Two box-counting algorithms have been employed. The first is a sorting algorithm, which is similar to the EBC algorithm described by BBS [2]. The second algorithm employs the agglomeration of results for a smallest set of boxes.

The following definitions are useful in the description of the box-counting algorithms.

(i) Let $S=\{s[j], j=1,\ldots,N\}$ be a subset of points on the fractal set, where, in order to simplify the exposition, the origin is chosen in such a way that all the $s_\delta[i]$ are positive.

(ii) Let $\{\mathbf{u}_\delta|\delta=1,d\}$ define an (arbitrary) orthogonal basis for the space containing S.

(iii) Define the largest scale:

$$E_{\max}=\max_{i,j,\delta}|\mathbf{u}_\delta\cdot(\mathbf{s}[i]-\mathbf{s}[j])| \ . \tag{4}$$

(iv) Map the fractal subset S into the unit hypercube (d cube) Q, by means of the affine transformation (which does not change its fractal properties):

$$\mathbf{s}^*[i]=\frac{\mathbf{s}[i]}{E_{\max}+\epsilon} \ , \tag{5}$$

where ϵ is a small positive number.

(v) Define the minimum normalized point set spacing

$$E_{\min}=\min_{i,j(i\neq j)}|\mathbf{s}^*[i]-\mathbf{s}^*[j]| \ . \tag{6}$$

The sorting box-counting algorithm

(i) Define a set of hypercubes (boxes), having edges E_m, whose logarithms are uniformly spaced:

$$E_m=\frac{1}{[R^{-1}(\epsilon^*)^{-m/M}]_{\text{int}}}, \quad m=0,1,\ldots,M \tag{7}$$

where $[\ ]_{\text{int}}$ indicates integer part.

The parameter R allows one to select normalized maximum box sizes less than unity. $R=1$ in EBC.

The parameter ϵ^* allows one to select minimum box edge values greater than the minimum normalized point set spacing $E_{\min}$. The minimum box edge is (approximately) $E_{\min}$ in EBC.

(ii) Sort the $\{\mathbf{s}^*[i]\}$ according to $s_1^*[i]$ using "INDEXED HEAPSORT"[4]. Computation time for this step goes like $N\log_{10}(N)$.

(iii) For each E_m compute the "partition function" $Z_m(q)$ as follows.

(a) Compute a set of integer-valued coordinates:

$$\{a_\delta[i]=s_\delta^*[i]/E_m|\delta=1,\ldots,d,i=1,\ldots,N\} \ . \tag{8}$$

N. b., the integer-valued coordinates will already be sorted according to $a_1[i]$.

(b) Define sublists of the $a_\delta[i]$ corresponding to the different values of $a_1[i]$.

(c) Compute the box-occupation probabilities $p_n(E_m)$ by sorting [4] the sublists via INDEXED HEAPSORT [4]. When the topological dimension $d>2$ sorting is accomplished by sequentially sorting or by sorting coordinates combined as in EBC, i.e., the $(d-1)N$ integer coordinates are combined into N integers comprising a "linear list" according to the prescription

$$\left\{f[i]|f[i]=\sum_{\delta=2}^{d}a_\delta[i]E_m^{2-\delta}, \ i=1,\ldots,N\right\} \ . \tag{9}$$

N.b., the length of the sublists will be substantially smaller than N unless E_m is near unity and when E_m is near unity the number of boxes is small and the sorting is much faster than $N\log_{10}(N)$.

(d) Compute the "partition function" $Z_m(q)$:

$$Z_m(q)=\sum_{n=1}^{N(E_m)}[p_n(E_m)]^q \ , \tag{10}$$

where $N(E_m)$ is the number of E_m boxes containing an element of S.

(iv) Compute the generalized box-counting fractal dimension $D_B(q)$, which is defined as the slope of the straight-line fit to $\log_{10}[Z_m(q)]/(q-1)$ vs $\log(E_m)$. As in BBS, the subscript B makes explicit the distinction between the intrinsic generalized dimension $D(q)$ and that measured by application of box counting to finite subsets of the given fractal set. The basis for this analysis is the generalized fractal dimension defined by Hentschel and Procaccia [5]:

$$D(q)=\frac{1}{q-1}\lim_{E_m\to 0}\frac{\log_{10}[Z_m(q)]}{\log_{10}(E_m)} \ , \tag{11}$$

where the limit is to be taken as E_m goes to zero for the perfect fractal. This form is also discussed in Halsey *et al.* [1].

Discussion of the sorting algorithm

Sorting algorithms are flexible in the range and distribution of box sizes employed and are efficient in the use of storage, however, computation time goes like $N\log_{10}(N)$ and precise coordinate definition is required for proper box allocation at each L. We discuss the present sorting algorithm by comparison with EBC.

Most of the features of EBC pertain to efficient use of computer storage and computation time and do not affect the reliability of the determined generalized fractal dimensions. In particular, although different sorting strategies may result in different computation times, they cannot affect the results.

There are five features of the EBC procedure which are not in accord with the present procedure and/or may not be routine assumptions in box counting and may influence the results.

(i) Although they do not disclose the essential features of the technique, BBS emphasize the fact that an essential feature of the EBC method is "classwise linear regression with random point selection" for the determination of the slope. "Classwise linear regression with random point selection" determines $D_B(q)$ for the subset ("pivotal values") of E_m (from the original logarithmically spaced set) for which "best" linear fitting is achieved.

The specific algorithm employed in EBC might make a substantive difference for $q<0$, where plots of $\log_{10}[Z_m(q)]/(q-1)$ against $\log_{10}(E_m)$ tend to be extensively scattered. [BBS did not report $D_B(q)$ for $q<0$.]

One expects that conventional linear regression is applied in most box-counting analyses. We have applied conventional linear regression based on minimization of least squares and absolute deviations in the present study.

(ii) EBC chooses the minimum box side in $\{E_m\}$ as the minimum magnitude of the vectorial distance between elements of the point set S. This prescription has the advantage of defining ϵ^* simply and uniquely in terms of the points in S. This assumption yielded unsatisfactory results when applied to the analysis of the multifractal point sets studied here.

(iii) EBC chooses the largest box size such that (within a small positive number ϵ) it just covers the point set in the arbitrarily selected orientation of the boxes. We have run calculations employing smaller "largest" E_m (i.e., we have run tests with $R<1$.). Although there are observable changes in $D_B(q)$ with R, the effects were small for $R\in[0.1,1]$ in the present work.

(iv) Uniform spacing of $\log(E_m)$ is prescribed in EBC. This amounts to a choice of weighting factors for different levels of scaling. We have investigated other sets of $\{E_m\}$, which seem to work just as well. However, logarithmically spaced $\{E_m\}$ is a natural choice in the sense that each order of scales is equally represented, and so it is adopted for the present work.

(v) The use of QUICKSORT [4] in EBC cannot lead to erroneous results and is usually the fastest sorting algorithm; however, it is dangerous in the sense that it can require computation times that go like N^2 (see Ref. [5]). Thus, we employ HEAPSORT [4], for which computation time always goes like $N\log_{10}(N)$.

Although sorting of the derived "linear list" as in EBC is conceptually simple, it seems to be a bad strategy (in terms of computation time) for large lists and ϵ^* near $E_{\min}$. For example, applying the present algorithm to a point set in $\mathbb{E}^1$, after the presort of the original (possibly floating point) list of coordinates, only *searching* of ordered lists of integers for changes is required, i.e., no further sorting is required in $\mathbb{E}^1$. Similar efficiencies associated with sorting shorter sublists can be expected for $d>1$. At some value of ϵ^*, sufficiently larger than $E_{\min}$, and/or for a small enough number of values of E_m, it becomes more efficient to avoid presorting the entire list even in $\mathbb{E}^1$, and one expects that no penalty in computation time should be associated with the use of the derived "linear list."

The agglomeration box-counting algorithm

(i) Define a set of nested hypercubes ("boxes") appropriate for the given (or anticipated) point set S, such that the boxes have edge lengths

$$E(k,m,\cdots)=E_0/(2^k\times3^m\ldots)\,,$$
$$k\in\{0,1,\ldots,K\},m\epsilon\{0,1,\ldots,M\},\ldots,\,. \tag{12}$$

We refer to the smallest boxes in the set, i.e., those having an edge length $E(K,M,\ldots)$, as elementary hypercubes or elementary boxes.

(ii) Determine the occupation numbers $\langle n_i\rangle$ for each of the elementary hypercubes. If the boxes do not include all members of S, redefine N:

$$N=\sum_{i}^{\text{elementary boxes}}\langle n_i\rangle\,.$$

(iii) For all combinations of $k,m,\ldots$, the following must occur.

(a) Compute the occupation numbers n_i for each of the $(2^k\times3^m\ldots)^d$ hypercubes by *summing* the occupation numbers of the constituent subsets of contained elementary hypercubes. We refer to this step as "agglomeration."

(b) Determine the $p_i=n_i/N$ for each box and compute the "partition function" $Z(k,m,\ldots;q)$ for the set of q for which $D_B(q)$ is to be determined as in Eq. (10). Empty boxes are not included.

(iv) Compute the generalized box-counting fractal dimension $D_B(q)$, which is defined as the slope of the straight-line fit to $\log[Z(k,m,\ldots;q)]/(q-1)$ vs $\log_{10}[E(k,m,\ldots)]$.

Discussion of the agglomeration algorithm

The agglomeration algorithm may be inefficient in its use of storage, since information about empty boxes is maintained. However, the method has definite advantages. It is well suited to application to "large" point sets, since computation time is essentially independent of N. Furthermore, the initial allocation of occupation values to the smallest boxes may be the natural way to treat the data obtained from automated image acquisition systems.

RESULTS

The present box-counting algorithms were applied to Euclidean point sets and point sets generated by simple Koch recursions [6] in $\mathbb{E}^1$ and $\mathbb{E}^2$. Applications of the sorting strategy in $\mathbb{E}^2$ were for sets having $1{,}000<N<50\,000$. The agglomeration technique was applied for $100<N<10^{11}$; sorting is not a practical alternative for the larger point sets.

Monofractals in $\mathbb{E}^2$

Integer dimensions; Euclidean point sets in $\mathbb{E}^2$

Euclidean point sets are thought of as degenerate examples of monofractals in the present investigation.

Random points on lines and on the unit square. Figure 1 shows fractal dimension values obtained by applying the agglomeration algorithm for $K=8$ and $M=1$ to a range of numbers of random points on a straight-line segment at two orientations in $\mathbb{E}^2$ with respect to the box axes. $D_B(q)$ converges at all q, but at negative q, $D_B(q)$ does not always converge to 1.0. Figure 2 shows fractal dimension values obtained by applying the agglomeration algorithm for $K=8$ and $M=1$ to a range of numbers of random points on the unit square. Convergence of $D_B(q)$ within 1% to 2.0 is obtained at $q=-5$, 0, 5, and 25;

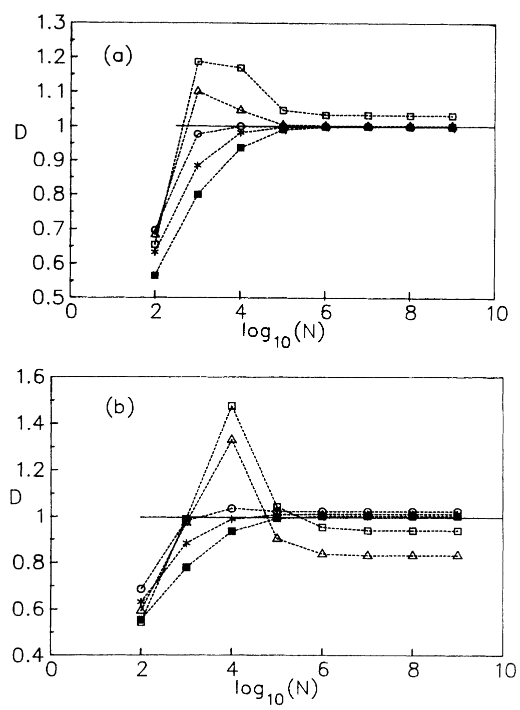

FIG. 1. Agglomeration box-counting [$K=8$, $M=1$] fractal dimension $D_B(q)$ vs the logarithm of the number of points for random points on line segments in $\mathbb{E}^2$. (a) Segment oriented at 45° with respect to the box edge. (b) Segment oriented at 18° with respect to the box edge. For all agglomeration box-counting figures; parameters are indicated as follows: $q=-25$ (open squares), $q=-5$ (triangles), $q=0$ (circles), $q=5$ (stars), $q=25$ (filled squares).

$D_B(-25)$ converges to 2.062.

Figure 3 shows sorting results for 2000 random points in the plane for $\epsilon^*=\{0.01,0.05,0.10,0.25\}$. The "best" values at negative q are obtained for $\epsilon^*\approx 0.05$ and at positive q for $\epsilon^*\approx 0.25$. Best values at all q would be obtained for ϵ^* between 0.05 and 0.10, which corresponds to an average box occupation of about ten points. Better convergence is obtained with larger N, but the existence of an optimal ϵ^*, substantially greater than the minimum spacing in S, is generally found.

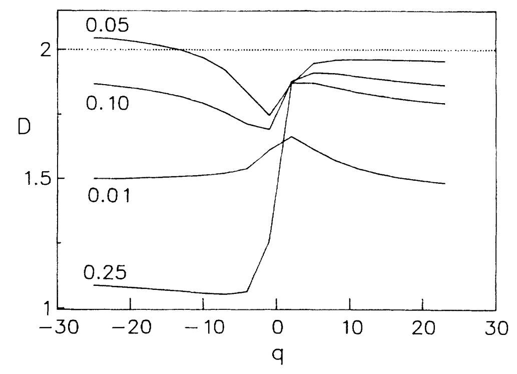

FIG. 3. Sorting box-counting fractal dimension $D_B(q)$ vs q for various ϵ^* for 2000 random points on the unit square in $\mathbb{E}^2$. Parameters are ϵ^* values for all sorting box-counting figures.

Similar results are obtained for random and equally spaced points on sawtooth curves and circular arcs.

Monofractal Koch constructions in $\mathbb{E}^2$

(Symmetric) Koch triadic snowflakes [6]. Figure 4 shows results obtained by applying the agglomeration technique for $K=8$, $M=1$ to Koch triadic snowflakes.

(i) Results at $q>0$. Although convergence is slower than that reported by BBS for EBC (on Euclidean point sets), the agglomeration $D_B(q)$ exhibits convergence from below with increasing N; $D_B(0)$, $D_B(5)$, and $D_B(25)$ are within 1% of $D(q)$ for $\log_{10}(N)>4$.

Sorting box-counting results for the triadic snowflake were similar to those obtained for *uniformly* spaced Euclidean curves in that for $2E_{\min}<\epsilon^*<0.05$, $0<q<9$ and $N>1{,}000$, the $D_B(q)$ are weakly ϵ^* dependent and are within about 2% of the intrinsic (constant) $D(q)$ values; $D_B(0)$ and $D_B(3)$ values within 0.5% of $D(0)$ were obtained for $\epsilon^*\approx 2E_{\min}$ and $N=3\times 4^7=49\,152$. $D_B(q)$ tended to decrease with increasing q for $q>5$.

(ii) Results at $q<0$. The values for $D_B(-25)$ and $D_B(-5)$ obtained by the agglomeration technique for $K=8$, $M=1$ do not converge for $N<10^9$; "best" values

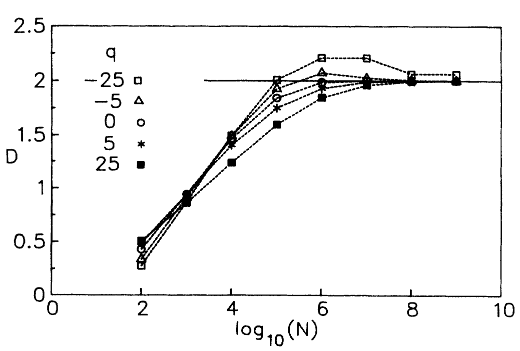

FIG. 2. Agglomeration box-counting $K=8$, $M=1$] fractal dimension $D_B(q)$ vs the logarithm of the number of points for random points on the unit square in $\mathbb{E}^2$.

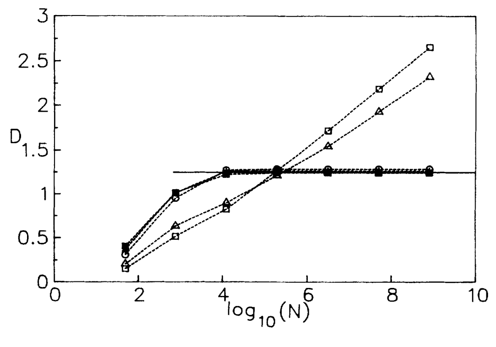

FIG. 4. Agglomeration box-counting [$K=8$, $M=1$] fractal dimension $D_B(q)$ vs the logarithm of the number of points for (symmetric) Koch triadic snowflakes in $\mathbb{E}^2$.

TABLE I. Normalized agglomeration box-counting [$K=8$, $M=1$] fractal dimension $D_B(q)/D(q)$ vs level for (symmetric) Koch triadic snowflakes in $\mathbb{E}^2$. ($N=3\times 4^{\text{level}}$).

Level	$q=-25$	$q=-5$	$q=0$	$q=5$	$q=25$
2	0.1187	0.1601	0.2435	0.2894	0.3191
4	0.4072	0.4993	0.7523	0.8004	0.8028
6	0.6511	0.7107	1.0070	0.9955	0.9688
8	1.0017	0.9581	1.0158	0.9985	0.9835
10	1.3632	1.2259	1.0161	0.9984	0.9834
12	1.7321	1.5283	1.0162	0.9983	0.9834
14	2.1031	1.8391	1.0162	0.9983	0.9834

are obtained when $\log_{10}(N)\approx 5.5$. The minimal box edge $E(8,1)=E_{\max}/(2^8\times 3)=E_{\max}/768$ corresponds to the minimum spacing at

$$\log_{10}(N)=\log_{10}(3)+D\log_{10}(768)\approx 4.1 .$$

Thus, for $q<0$ the best values were obtained for the minimal box edge about ten times larger than the minimal spacing in the point set. The $D_B(-5)$ and $D_B(-25)$ curves cross near $\log_{10}(N)=4.1$. Table I gives values of normalized $D_B(q)$ [$=D_B(q)\log_{10}(3)/\log_{10}(4)$] vs level ($N=3\times 4^{\text{level}}$) and q for the Koch triadic snowflakes.

For $q<0$, $D_B(q)$ curves obtained by sorting were strongly dependent on ϵ^* and tended to more closely resemble the intrinsic (constant) $D(q)$ for $\epsilon^*\approx 2E_{\min}$. For $\epsilon^*\leq E_{\min}$, $D_B(q)$ are monotonic *increasing* at negative q. Similar results are obtained for Koch quadric islands [6].

Multifractals in $\mathbb{E}^1$

The sorting box-counting algorithm was applied to the logistic equation mapping employing one-dimensional hypercubes. The $f-\alpha$ spectra deduced from $D_B(q)$ were in accord with those shown in Tobochnik and Gould [3].

Multifractals in $\mathbb{E}^2$

The box-counting algorithms were applied to asymmetric Koch triadic snowflakes and Koch split snowflake halls. The asymmetric Koch triadic snowflakes were produced by modifying the symmetric Koch triadic snowflake generator such that the angles are unchanged but the outer segments are twice as large as the central segments, i.e., the inner and outer segments of the generator have relative lengths 0.2 and 0.4, respectively.

Mandelbrot [6] defines split snowflake halls as the level-4 form of the construction, which at level 3, he defines as the monkey tree. Here we refer to all levels of the construction as split snowflake halls.

Asymmetric Koch snowflakes

Sorting results. Figure 5 shows $D_B(q)$ obtained by sorting box-counting for level-7 ($N=49,152$), asymmetric [0.4,0.2] Koch snowflakes. The dashed curve is the intrinsic $D(q)$. The following attributes may be observed.

(i) Results for $5>q>0$. For ϵ^* values ranging between about 0.001 ($\approx 75E_{\min}$) and about 0.010 ($\approx 750E_{\min}$), $D_B(q)$ closely approximates the intrinsic $D(q)$. $D_B(0)$ values are weakly ϵ^* dependent over the specified range of ϵ^* and are within 1% of their intrinsic values. $D_B(3)$ values are weakly ϵ^* dependent over the specified range of ϵ^* and are generally within 2% of their intrinsic values. The discrepancy is larger at larger q. The discrepancy between $D_B(q)$ and $D(q)$ is quite large for $\epsilon^*=E_{\min}$.

The requirement that $\epsilon^*>75E_{\min}$ can be understood as follows. $E_{\min}$ is determined by the most dense regions of the point set. The snowflake considered has the least

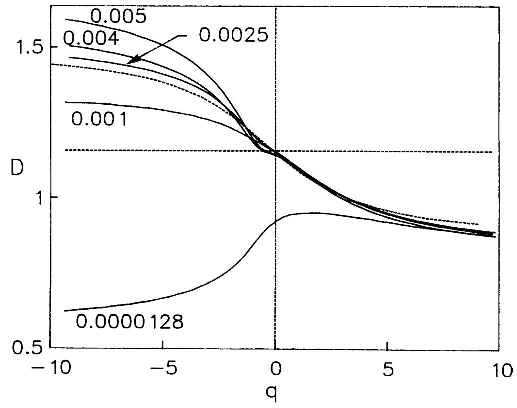

FIG. 5. Sorting box-counting fractal dimension $D_B(q)$ vs q for various ϵ^* for asymmetric [0.4, 0.2] Koch snowflakes in $\mathbb{E}^2$. $N=49\,152$. The dashed curve is the analytic result.

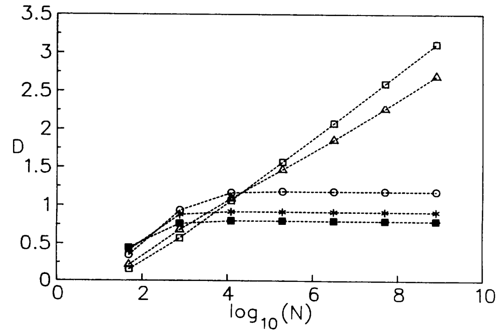

FIG. 6. Agglomeration box-counting [$K=8$, $M=1$] fractal dimension $D_B(q)$ vs the logarithm of the number of points for asymmetric [0.4, 0.2] Koch snowflakes in $\mathbb{E}^2$.

TABLE II. Normalized agglomeration box-counting [$K=8$, $M=1$] fractal dimension $D_B(q)/D(q)$ vs level for asymmetric [0.4, 0.2] Koch triadic snowflakes in $\mathbb{E}^2$. ($N=3\times4^{\text{level}}$).

Level	$q=-25$	$q=-5$	$q=0$	$q=5$	$q=25$
2	0.1029	0.1548	0.2986	0.4298	0.5075
4	0.3862	0.4917	0.8062	0.9193	0.8707
6	0.7162	0.7859	1.0053	0.9577	0.9113
8	1.0580	1.0588	1.0210	0.9581	0.9115
10	1.4027	1.3428	1.0220	0.9580	0.9115
12	1.7496	1.6372	1.0221	0.9580	0.9115
14	2.0992	1.9445	1.0221	0.9580	0.9115

dense regions, which are $(0.4/0.2)^7=128$ times less dense. Therefore, $\epsilon^*=128E_{\min}$ are minimal spacings in the least dense regions of the level-7 construct. Hence, $\epsilon^*=75E_{\min}$ is a reasonable value to characterize the entire point set and it is clear that the EBC choice $\epsilon^*=E_{\min}$ is too small.

(ii) Results at $q<0$. $D_B(q)$ curves were strongly dependent on ϵ^*. Curves that approximate $D(q)$ reasonably well over the entire range of q are obtained for ϵ^* between 0.001 and 0.004. The curve obtained for ϵ^* = 0.0025 is closest to $D(q)$ while that for 0.001 is smoother near $q=0$ but is slightly farther from $D(q)$. One may observe the beginning of the formation of a slight belly near $q=0$, as ϵ^* increases through 0.0025.

Agglomeration results. Figure 6 shows $D_B(q)$ curves for $q=\{-25,-5,0,5,25\}$ and various N for the [0.4,0.2] asymmetric triadic snowflakes measured via $K=80$, $M=1$ boxes; Table II gives values of normalized $D_B(q)$ [$=D_B(q)/D(q)$] vs level ($N=3\times4^{\text{level}}$) and q. Converged values for $q\geq0$ are obtained as 0.4^{level} passes through the box edge length $E(8,1)$ of the elemental boxes. "Best" values at negative q occur for these transitional cases, which falls at $\log_{10}(N)$ slightly beyond that for which the $D_B(-5)$ and $D_B(-25)$ curves cross.

Figure 7 shows similar results for $K=3$, $M=1$, $P=1$ [$E(3,1,1)=E_{\max}/(2^3\times3\times5)=E_{\max}/120$]; Table III gives values of normalized $D_B(q)$ vs level and q. Note, in particular, that convergence is obtained for values of N about $\frac{1}{5}$th as large as in the $K=8$, $M=1$ case.

Split snowflake halls

Sorting results. Figure 8 shows $D_B(q)$ obtained by sorting box-counting for level-4 ($N=1+11^4$) split snowflake halls. The dashed curve is the intrinsic $D(q)$. The following features may be observed:

(i) Results for $q>3$. For ϵ^* values ranging between about $7E_{\min}$ and $30E_{\min}$, $D_B(q)$ closely approximates the intrinsic $D(q)$. $D_B(3)$ values are weakly ϵ^* dependent over the specified range of ϵ^*, but are generally within 2% of their intrinsic values and are closer at larger q. The discrepancy between $D_B(q)$ and $D(q)$ is large for $\epsilon^*=E_{\min}$.

(ii) Results for $3>q>-1$. For ϵ^* values ranging between about $7E_{\min}$ and 0.07, $D_B(q)$ are weakly ϵ^* dependent, but lie substantially below the intrinsic $D(q)$. $D_B(0)$ values are about 12% low.

(iii) Results at $q<-1$. $D_B(q)$ curves were strongly dependent on ϵ^*. Curves that most closely approximated $D(q)$ for this range of q, in the sense of rms deviation, behaved badly near $q=0$ and had ϵ^* values outside the range and larger than those for which "best" approximation in the $q>-1$ region obtains. No value of ϵ^* produced a monotonic decreasing $D_B(q)$.

TABLE III. Normalized agglomeration box-counting [$K=3$, $M=1$, $P=1$] fractal dimension $D_B(q)/D(q)$ vs level for asymmetric [04., 0.2] Koch triadic snowflakes in $\mathbb{E}^2$. ($N-3\times4^{\text{level}}$).

Level	$q=-25$	$q=-5$	$q=0$	$q=5$	$q=25$
1	0.0543	0.0824	0.1438	0.2630	0.3497
2	0.1834	0.2667	0.5045	0.6941	0.7652
3	0.3949	0.5077	0.8244	0.9155	0.8637
4	0.6178	0.7079	0.9773	0.9646	0.9083
5	0.8437	0.8819	1.0177	0.9737	0.9177
6	1.0930	1.0836	1.0246	0.9750	0.9198
7	1.3419	1.2949	1.0258	0.9753	0.9201
8	1.5448	1.4492	1.0261	0.9751	0.9199
9	1.7295	1.6108	1.0263	0.9751	0.9199
10	1.8953	1.7611	1.0263	0.9751	0.9199
11	2.0597	1.9090	1.0263	0.9751	0.9199
12	2.2287	2.0656	1.0263	0.9751	0.9199
13	2.3981	2.2227	1.0263	0.9751	0.9199
14	2.5674	2.3795	1.0263	0.9751	0.9199

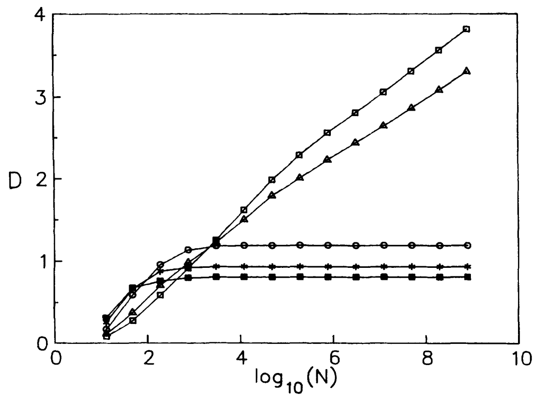

FIG. 7. Agglomeration box-counting [$K=3$, $M=1$, $P=1$] fractal dimension $D_B(q)$ vs the logarithm of the number of points for asymmetric [0.4, 0.2] Koch snowflakes in $\mathbb{E}^2$.

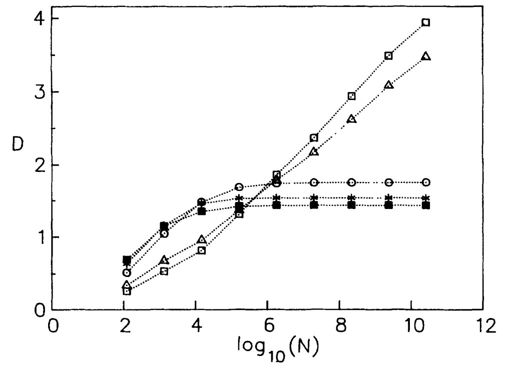

FIG. 9. Agglomeration box-counting [$K=8$, $M=1$] fractal dimension $D_B(q)$ vs the logarithm of the number of points for split snowflake halls in $\mathbb{E}^2$.

Agglomeration results. Figure 9 shows $D_B(q)$ curves for $q=\{-25,-5,0,5,25\}$ and various N for split snowflake halls measured via $K=8$, $M=1$ boxes; Table IV gives values of normalized $D_B(q)$ [$=D_B(q)/D(q)$] vs level ($N=1+11^{\text{level}}$) and q. "Best" values at negative q occur for $\log_{10}(N)$ slightly larger than that for which the $D_B(-5)$ and $D_B(-25)$ curves cross.

Figure 10 shows similar results for $K=3$, $M=1$, $P=1$; Table V gives values of normalized $D_B(q)$ vs level and q. Note, in particular, that convergence is obtained for values of N about $\frac{1}{5}$th as large as in the $K=8$. $M=1$ case.

CONCLUSIONS

The sorting and agglomeration box-counting algorithms are effective for the determination of $D_B(q)$ for $q \geq 0$. However, the present implementations yield unreliable results for $q<0$.

$D_B(q)$ values converged from below for $q \geq 0$ and overshot and converged from above at $q<0$ for random points on Euclidean objects. Therefore, convergence to $D(q)$ for Euclidean point sets is not a sufficient condition for demonstrating the effectiveness of an algorithm.

Sorting box-counting algorithms require storage of $d \times N$ double-precision (eight-byte) floating-point numbers in E^d and CPU time varies as $N \log_{10}(N)$. The input data to the algorithm are the coordinates of the points. In order to achieve best results, the particle coordinates must be known precisely so that box occupancies for the various box edge lengths in the set can be determined; this constraint is relatively easy to satisfy for mappings or other recursively defined point sets, but may be exacting for experimentally acquired point set coordinates.

The agglomeration box-counting algorithm requires storage of $(2^K \times 3^M \cdots)^d$ integers and CPU time is proportional to $(2^K \times 3^M \cdots)^d$. Although we used four-byte integers in the present work, two-byte integers are probably sufficient in practice. CPU time and storage requirements are independent of N and therefore agglomeration algorithms may be the only practical technique for application to cases where N is "large." The input data to the algorithm are elementary box occupation numbers rather than precise coordinates. This could be an important distinction for image acquisition systems, which provide data in this form. Thus, the agglomeration technique works equally well for mappings or other recursively

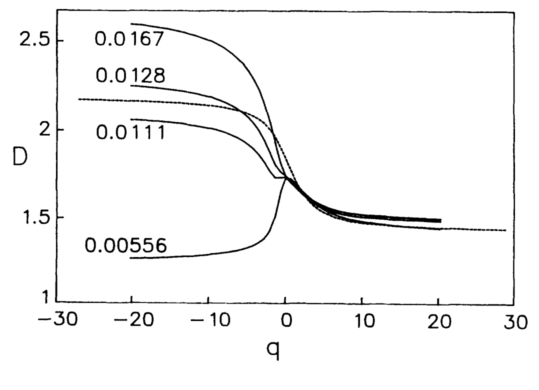

FIG. 8. Sorting box-counting fractal dimension $D_B(q)$ vs q for various ϵ^* for level-4 ($N=14\,642$) split snowflake halls in $\mathbb{E}^2$. The dashed curve is the analytic result.

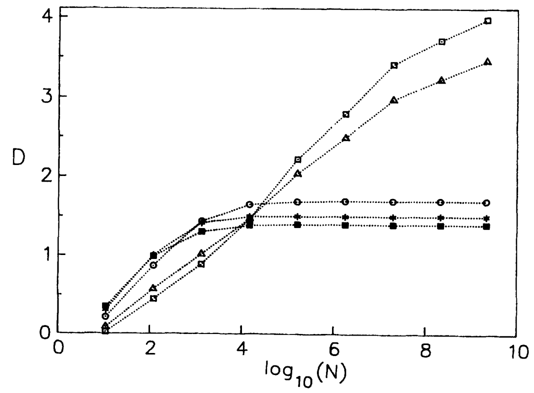

FIG. 10. Agglomeration box-counting [$K=3$, $M=1$, $P=1$] fractal dimension $D_B(q)$ vs the logarithm of the number of points for split snowflake halls in $\mathbb{E}^2$.

TABLE IV. Normalized agglomeration box-counting $K=8$, $M=1$] fractal dimension $D_B(q)/D(q)$ vs level for split snowflake halls in $\mathbb{E}^2$. ($N=1+11^{\text{level}}$).

Level	$q=-25$	$q=-5$	$q=0$	$q=5$	$q=25$
2	0.1203	0.1628	0.2747	0.4068	0.4678
3	0.2464	0.3226	0.5576	0.7343	0.7767
4	0.3764	0.4559	0.7908	0.9319	0.9135
5	0.6045	0.6526	0.9022	0.9748	0.9609
6	0.8599	0.8533	0.9303	0.9772	0.9688
7	1.0927	1.0349	0.9352	0.9771	0.9689
8	1.3575	1.2493	0.9366	0.9771	0.9689
9	1.6121	1.4718	0.9370	0.9771	0.9689
10	1.8223	1.6595	0.9372	0.9771	0.9688

TABLE V. Normalized agglomeration box-counting $K=3$, $M=1$, $P=1$] fractal dimension $D_B(q)/D(q)$ vs level for split snowflake halls in $\mathbb{E}^2$. ($N=1+11^{\text{level}}$).

Level	$q=-25$	$q=-5$	$q=0$	$q=5$	$q=25$
1	0.0091	0.0401	0.1128	0.1931	0.2338
2	0.2064	0.2742	0.4645	0.6393	0.6693
3	0.4098	0.4846	0.7682	0.9003	0.8818
4	0.6772	0.7033	0.8799	0.9531	0.9378
5	1.0265	0.9744	0.9016	0.9579	0.9490
6	1.2935	1.1909	0.9061	0.9571	0.9478
7	1.5803	1.4236	0.9070	0.9572	0.9479
8	1.7172	1.5409	0.9073	0.9573	0.9480
9	1.8404	1.6545	0.9075	0.9573	0.9480

defined point sets and for experimentally acquired point sets.

The values of N required for convergence of D at $q \geq 0$ were approximately the same for the sorting and agglomeration box-counting algorithms in the present investigation.

For example, for $d=2$ and $D(0) < 1.5$, our experience indicates that $N \approx 10^4$ is required for convergence. In this case, storage requirements for ($K=3$, $M=1$, $P=1$) agglomeration (four-byte integers) is approximately 60 kbytes while sorting requires 200 kbytes. Larger N and number of elementary boxes are required for larger $D(0)$, but the relative advantage of the agglomeration technique with regard to storage is maintained.

We have not tested convergence of box-counting techniques for $d>2$, however we anticipate that substantially larger N will be required for convergence. One might speculate that for $d=3$, $N \approx (10^4)^{1.5} = 10^6$, agglomeration ($K=3$, $L=1$, $M=1$) uses 7 Mbytes. Sorting would require 25 Mbytes. Of course, if convergence is obtained for $N \approx 10^4$ for $d=3$, then sorting would require only 0.25 Mbytes while agglomeration would still require 7 Mbytes.

Sorting box-counting algorithms might be the best choice for relatively small sets of precisely defined points. If "classwise linear regression with random point selection" implicitly yields sets of E_m which give good values for $D_B(q)$ at $q<0$, then EBC is probably best for the cases to which it may be applied. However, if this is the case, the principle can probably be applied to refine the agglomeration technique as well.

Agglomeration box counting resulted in values for $D_B(q)$ at least as good as those determined by sorting for all cases studied here. For the large point sets employed in our convergence studies, sorting would have been (for the present) impractical. Computation times for the moderately sized point sets (required for convergence) would be competitive; agglomeration, of course, becomes relatively more efficient as N increases.

[1] Thomas C. Halsey, Mogens H. Jensen, Leo P. Kadanoff, Itamar Procaccia, and Boris I. Shraiman, Phys. Rev. A **33**, 1141 (1986).

[2] A. Block, W. von Bloh, and H. J. Schellnhuber, Phys. Rev. A **42**, 1869 (1990). Other recent references to sorting algorithms for the determination of $D(0)$ include: L. S. Liebovitch and T. Toth, Phys. Lett. A **141**, 386 (1989); X. J. Hou, R. G. Gilmore, G. B. Mindlin, and H. G. Solari, Phys. Lett. A **151**, 43 (1990).

[3] Harvey Gould and Jan Tobochnik, Comput. Phys. **4**, 202 (1990).

[4] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling, *Numerical Recipes* (Cambridge University Press, Cambridge, 1986).

[5] H. G. E. Hentschel and I. Procaccia, Physica D **8**, 435 (1983).

[6] B. B. Mandelbrot, *Fractal Geometry of Nature* (Freeman, New York, 1983).

[7] For example, B. B. Mandelbrot, C. J. G. Evertsz, and Y. Hayakawa, Phys. Rev. A **42**, 4528 (1990).