Geophys. J. Int. (2000) **142,** 108–116

Measuring the fractal dimensions of ideal and actual objects: implications for application in geology and geophysics

Guido Gonzato, Francesco Mulargia and Matteo Ciccotti
Dipartimento di Fisica, Settore di Geofisica, Università di Bologna, Italy. E-mail: guido@ibogfs.df.unibo.it

Accepted 2000 February 10. Received 2000 January 31; in original form 1999 July 26

SUMMARY

The box-counting algorithm is the most commonly used method for evaluating the fractal dimension D of natural images. However, its application may easily lead to erroneous results. In a previous paper (Gonzato *et al.* 1998) we demonstrated that a crucial bias is introduced by insufficient sampling and/or by uncritical application of the regression technique. This bias turns out to be common in many practical applications. Here it is shown that an equally important additional bias is introduced by the orientation, placement and length of the digitized object relative to that of the initial box. Some additional problems are introduced by objects containing unconnected parts, since the discontinuities may or may not be indicative of a fractal pattern. Last, but certainly not least in magnitude, the thickness of the digitized profile, which is implicitly controlled by the scanner resolution versus the image line thickness, plays a fundamental role. All of these factors combined introduce systematic errors in determining D, the magnitudes of which are found to exceed 50 per cent in some cases, crucially affecting classification. To study these errors and minimize them, a program that accounts for image digitization, zooming and automatic box counting has been developed and tested on images of known dimension. The code automatically extracts the unconnected parts from a digitized shape given as input, zooms each part as optimally as possible, and performs the box-counting algorithm on a virtual screen. The size of the screen can be set to meet the sampling requirement needed to produce stable and reliable results. However, this code does not provide image vectorization, which must be performed prior to running this program. A number of image vectorizing codes are available that successfully reduce the thickness of the image parts to one pixel. Image vectorization applied prior to the application of our code reduces the sampling bias for objects with known fractal dimension to around 10–20 per cent. Since this bias is always positive, this effect can be readily compensated by a multiplying factor, and estimates of the fractal dimension accurate to about 10 per cent are effectively possible.

Key words: box-counting algorithm, fractal dimension.

1 INTRODUCTION

The application of fractal analysis on 2-D maps, representing actual shapes or patterns, provides an important quantitative classification tool, but it is riddled with practical problems. These problems are commonly associated with the evaluation of the fractal dimension D and the range of scale invariance R. In a previous paper (Gonzato *et al.* 1998) we have shown that many of the results presented in the literature are marred by a faulty application of linear regression in the *box-counting* algorithm, which will be briefly summarized in the next section. The most common error is insufficient sampling, which makes it impossible to attach real significance to the calculated value. Furthermore, providing too narrow a bandwidth of observation R gives little physical meaning to the results.

To overcome these common difficulties, we have implemented a computer program, VSBC, available from our web site (see Appendix A). It uses a large virtual screen (4096 pixels; this can be increased) on which digitized maps can be loaded and performs the box-counting algorithm automatically. However, further investigation has shown that there are other important causes of bias to be aware of. In the following sections these potential problems are discussed, and a software-based solution is provided.

2 THE BOX-COUNTING METHOD

The box-counting algorithm comprehends different definitions of 'dimension' (Mandelbrot 1967, 1982; Grassberger 1993). To calculate D, one covers the object with a grid of squares (or, in case of 3-D surfaces, of cubes) initially of side η_1 and then counts the number N_1 of squares that include part of the object. The measurement is then carried out using side η_2, obtaining N_2 squares. This step is repeated M times, using squares of increasingly shorter side. It can be shown (Mandelbrot 1982) that

$$N \propto \eta^{-D}, \qquad (1)$$

whence

$$\log(N(\eta)) = a - D\log(\eta). \qquad (2)$$

One calculates the regression line between the independent variable $\log(\eta_i)$ and the dependent variable $\log(N(\eta_i))$, where $i=1, \ldots, M$. D is given by the absolute value of the line slope. An adequate number of steps (at least 15–20, say) is necessary for credible application of the linear regression technique. In addition, close attention should be paid to the range over which the box counting is applied, because this appears to be tied to the processes that lead to the scale invariance, which, over limited ranges, may also be a result of pure randomness (Hamburger *et al.* 1996; Malcai *et al.* 1997). There are several sources of bias in this procedure, notably the finiteness of the sampled shape and its irregular geometry (Ouillon & Sornette 1996). It should be added that some geological situations (for example, sediments covering fault lines; Ouillon *et al.* 1996) may lead to erroneous results.

In the following examples of box-counting analysis, we shall employ a division step equal to 2. As an immediate consequence, it is highly desirable that the initial box side be a power of 2.

3 ERRORS AFFECTING THE BOX-COUNTING PROCEDURE

When one digitizes a shape or a map for subsequent fractal analysis, the way the image is generated by the scanner is usually disregarded as a trivial detail. However, we shall show that the length, orientation and placement of an image with respect to the initial box are all potential causes of error that can propagate into significant bias in the estimate of D. This is important since the placement of the initial box is normally either arbitrary or determined by field exposure (see Turcotte 1989). Here we assume an ideal case of 100 per cent exposure, that is, maps that may be obtained from rocky desert areas. Moreover, the presence of unconnected parts in the same image must be accounted for, since these parts do not necessarily belong to the same higher-order shape. Finally, the process of digitization itself is a major cause of error, as we show in Section 3.5.

3.1 Geometrical constraints

To begin with, we note that the box-counting algorithm yields an image's correct value of D only for special geometrical arrangements. Let us consider an example in which we perform the box-counting procedure on a line segment. One obtains $D=1$ if and only if (1) both the box side and the line have a side length equal to a power of 2 and (2) the line is either vertical or horizontal. We discuss point (1) in Section 3.2 and point (2) in Section 3.3.

3.2 Shape position

We note first of all that with a side length equal to a power of 2 the box side halves at each step, while the number of full sub-boxes doubles. The optimal arrangement is shown in Fig. 1, where the lower line satisfies the above conditions, and the line above breaks them; the number of full sub-boxes will not double exactly, thus affecting the calculated value of D. An application of the optimal arrangement, involving an initial box side and line length equal to 4096 pixels, produces the data shown in Table 1(a); the resulting regression line is $\log(N(\eta)) = 8.318 - 1.000\log(\eta)$. Making the line just 1 pixel shorter, thus breaking condition (1), D becomes 0.97 (see Table 1b). This bias is always negative and is limited to a few per cent.

Unfortunately, a common procedure in the literature is to use an initial box side that is both different from a power of 2 and, more importantly, larger than the shape it contains. Such a move introduces additional negative bias in the estimate of D, owing to the empty boxes that surround the image; we call these the 'over-empty' boxes. In fact, during the box-counting procedure an arbitrary number of over-empty boxes will also be counted, contributing in a systematic way to the final value of D.

To make this point clearer, let us consider three images framed in a 1024-pixel-long bounding box: (i) a single pixel; (ii) a 1024-pixel-long line; and (iii) a 200-pixel-long line. What calculated dimension should be expected for the shorter line? This value will obviously be greater than 0, but also less than 1 owing to the contribution of the over-empty boxes. One might incorrectly conclude that a fractal has been found.

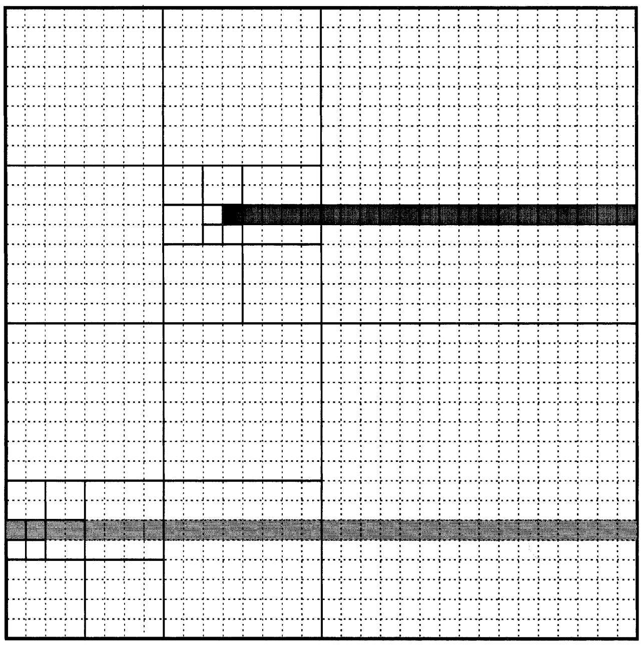

Figure 1. Bounding box containing a line of correct width (light grey) and a line segment (dark grey) that causes over-empty boxes.

Table 1. (a) Data obtained from the application of the box counting on a 4096-pixel-long horizontal line, framed in a box of equal side. The resulting regression line is $\log(N(\eta)) = 8.318 - 1.000 \log(\eta)$. (b) Data obtained from the application of the box counting on a 4095-pixel-long horizontal line, framed in a box of equal side. The resulting regression line is $\log(N(\eta)) = 8.265 - 0.972 \log(\eta)$. η_i = box side at step i; $N(\eta_i)$ = number of boxes containing part of the image at step i.

	η_i	$N(\eta_i)$	$\log(\eta_i)$	$\log(N(\eta_i))$
(a)	4096	1	8.317766	0.000000
	2048	2	7.624619	0.693147
	1024	4	6.931472	1.386294
	512	8	6.238325	2.079442
	256	16	5.545177	2.772589
	128	32	4.852030	3.465736
	64	64	4.158883	4.158883
	32	128	3.465736	4.852030
	16	256	2.772589	5.545177
	8	512	2.079442	6.238325
	4	1024	1.386294	6.931472
	2	2048	0.693147	7.624619
	1	4096	0.000000	8.317766
(b)	4095	1	8.317522	0.000000
	2047	3	7.624131	1.098612
	1023	5	6.930495	1.609438
	511	9	6.236370	2.197225
	255	17	5.541264	2.833213
	127	33	4.844187	3.496508
	63	65	4.143135	4.174387
	31	133	3.433987	4.890349
	15	273	2.708050	5.609472
	7	585	1.945910	6.371612
	3	1365	1.098612	7.218910
	1	4102	0.000000	8.319230

We tested this effect in two ways. First, we shifted a 2048-pixel-long line, one pixel at a time, along the side of a 4096-pixel-wide initial box. The resulting fractal dimension D varies moderately; errors of up to -9 per cent are possible. Similar results are obtained by shifting a Koch curve relative to the initial box (Fig. 2). We then evaluated D for straight lines of lengths varying from 64 to 4096 pixels, and repeated this test using the Koch curve. The results are very similar: in both cases D increases from 0 to the correct value as the length of the line approaches the bounding box side. The results for the Koch curve are shown in Fig. 3.

It is easy to find a remedy for this cause of error: one should simply ensure that the bounding box side coincides with the width of the shape. We therefore implemented a zooming capability in our code; essentially, we frame the larger box so that the shape touches its upper and left edges (Fig. 4). Zooming the image solves the over-empty boxes effect.

We verified that the shifted line and Koch curve when applying the zooming gave measured values of D obviously much closer to 1.00 for the line, and to 1.35 for the Koch curve. The latter value does not coincide with the theoretical D because the calculation is affected by another source of bias, which will be explained in the following section. We also repeated the tests on the variable-length line and the Koch curve. Applying the zooming, one removes a significant fraction of the bias; the errors on D for the line are below -4 per cent; the results for the Koch curve are shown in Fig. 5. Note that in this case the remaining error is due to the initial box side not being a power of 2.

The zooming technique makes it possible (and advisable) to employ a different division step. Instead of 2, the initial box side can be divided by $\sqrt{2}$, doubling the number of data points

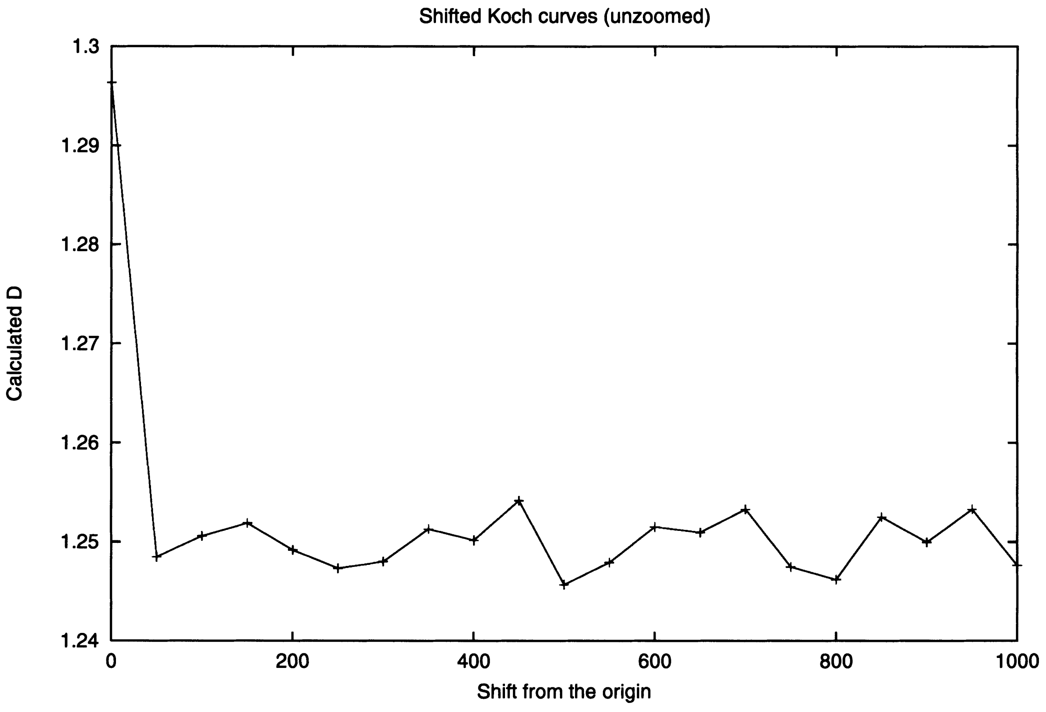

Figure 2. Calculated D for a 1024-pixel-long Koch curve, shifted from the lower-left corner of the bounding box.

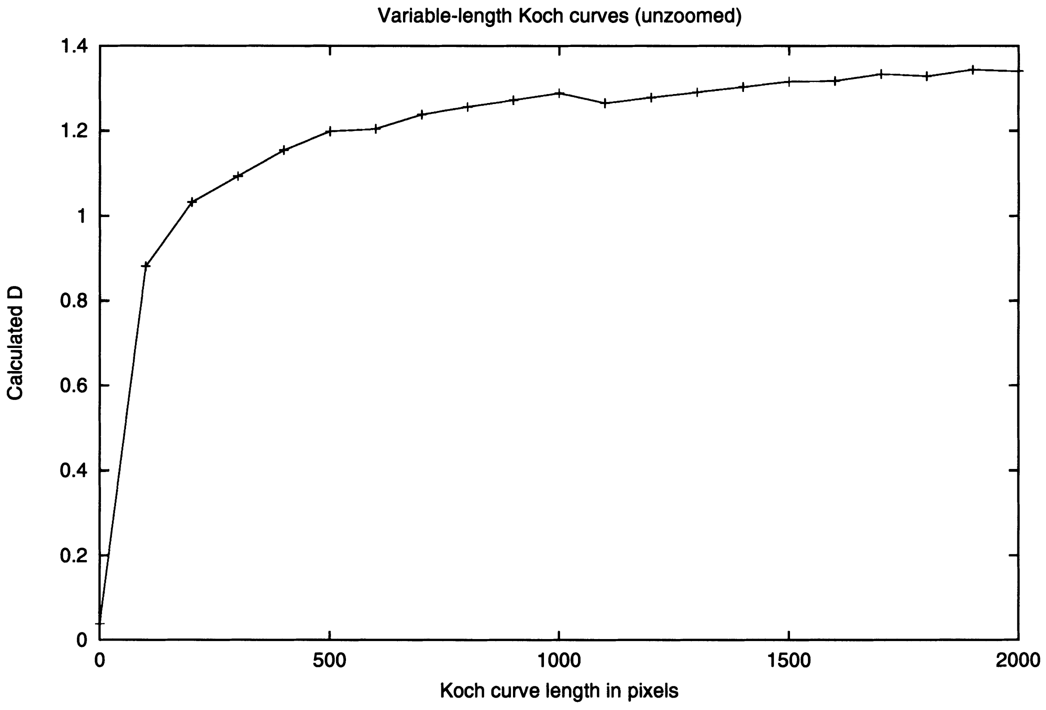

Figure 3. Calculated D for Koch curves ranging in length from 1 to 2000 pixels.

and increasing the scope of the regression analysis (see Gonzato *et al.* 1998). Dividing the box side by $\sqrt{2}$, though, introduces a finite area that cannot be covered by the boxes. We call this unaccounted-for area the 'remainder', which diminishes with the number of boxes during the iteration. We verified that the error introduced by the remainder is negligible compared to that introduced by the outlying boxes.

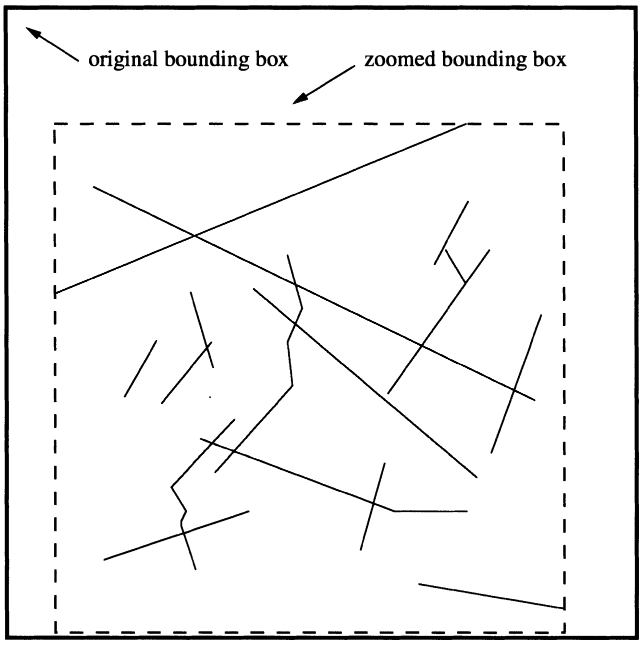

Figure 4. Bounding box redefined to coincide with the upper and left-hand sides of the shape.

In the examples outlined in the following sections, application of the zooming technique is implicitly assumed.

3.3 Shape angle

Having examined the effects of position and bounding box side, we proceeded to study the effect of the angle of the shape relative to the initial box. From the experience gathered with the shifted line, one easily foresees that analogous effects are bound to happen. In fact, it is obvious that the number of sub-boxes containing part of a horizontal line will change if the line is tilted by any angle, as shown in Fig. 6. Any angle different from 0° or 90° will change the number of full sub-boxes for two reasons: (i) because of geometrical considerations and (ii) because a digitized tilted line is, in most cases, approximated by a 'staircase' with double pixels at most steps. These double pixels cause a higher number of sub-boxes to be full than is necessary; we shall call these 'overfull' boxes. This effect, brought about by the algorithms employed in digitization, is not removable.

In Fig. 7 the effects of tilting a line from 0 to $\pi/4$ are shown; the resulting values of D are affected by bias that is apparently constant (due to the 'staircase effect' mentioned above) and on average 6–8 per cent above the true value.

These effects cannot be completely eliminated. An immediate consequence is the impossibility of obtaining correct values of D for complex shapes such as the Koch curve. In fact, any realistic image will contain several parts, each with a different inclination with respect to the bounding box. As a result, it is not possible to remove this source of bias completely by placing the map parallel to the side of the scanner, unless the map contains only lines at 0° to the axes.

We performed the same simulation as above but tilting a Koch line from 0 to $\pi/4$; similar effects are produced (Fig. 8).

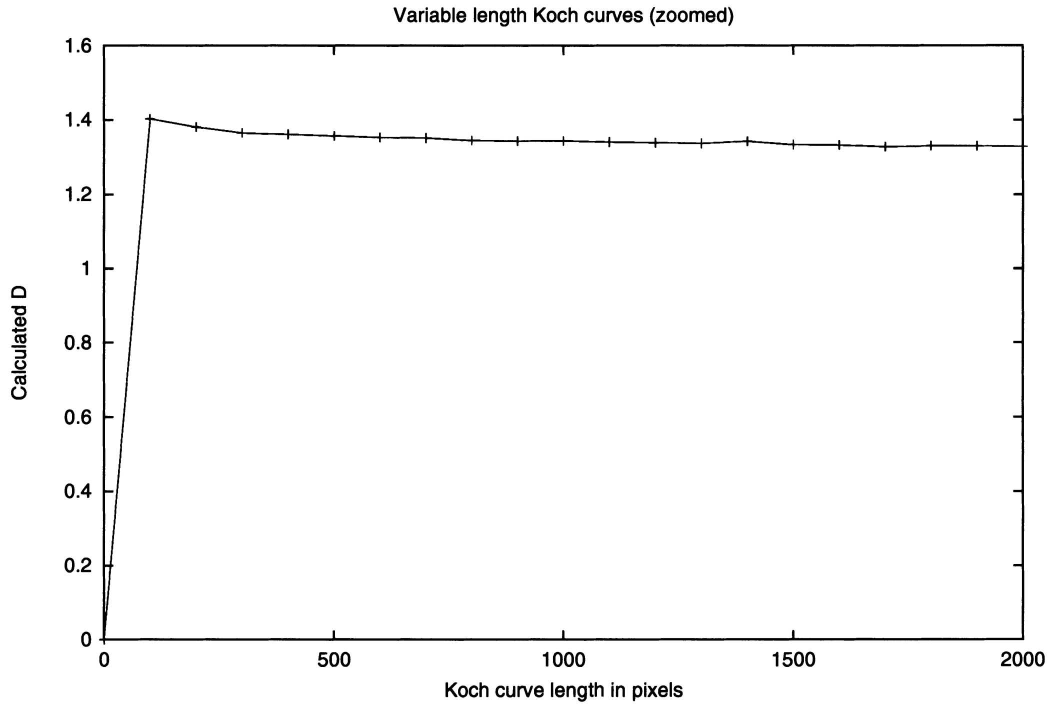

Figure 5. Calculated *D* for Koch curves ranging in length from 1 to 2000 pixels; the image has been zoomed.

Unlike the case of the straight line, in which the best results are obtained if the inclination is 0, the Koch curve does not show a particular configuration for which the exact value of *D* is reached. Automatic correction of this cause of error would only be possible for simple shapes such as parallel lines (by rotating the image before performing the box counting), but it is clearly not feasible for general images.

In conclusion, any real map will be characterized by a calculated *D* that is affected by unremovable positive bias due to the relative orientations of the image and the initial box. Quantifying this bias exactly is not possible, but some average estimate can be attempted.

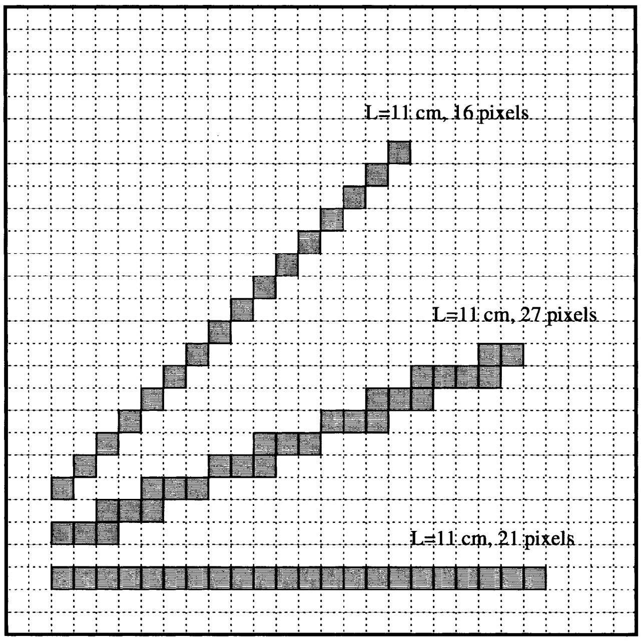

Figure 6. Digitized straight lines, horizontal and tilted. Note how the number of pixels is different in the three lines, affecting the box-counting results.

3.4 Estimating the digitization bias

When one digitizes a map with a scanner, a shape consisting of clusters of square pixels with random positions and angles is produced. One can therefore get an idea of the digitization bias by averaging it over a number of measurements for known objects. In order to determine the bias, we ran a simulation generating 100 maps, each representing a line with random length, angle and placement on a 4096-pixel-wide screen. The same procedure was repeated using the Koch curve; the results are shown in Fig. 9. In both cases, the average bias introduced by the angle is +11 per cent. There is some tendency of the overfull boxes to compensate for the over-empty ones, but the results are unpredictable. It is thus convenient to correct the over-empty boxes effect first, then the overfull boxes effect. Note that if zooming had not been applied, the average bias would have been equal to −24 per cent for the random lines and −38 per cent for the Koch curves. In general, according to our extensive testing, a positive bias of 10–20 per cent is obtained on zoomed shapes.

The application of zooming also greatly reduced the variability between the different realizations, thus increasing the reliability of results produced by one (or a few) measurement. Therefore, decreasing the calculated value of *D* by 10–20 per cent is expected to provide the best possible estimate of *D*, within 5–10 per cent of the real value.

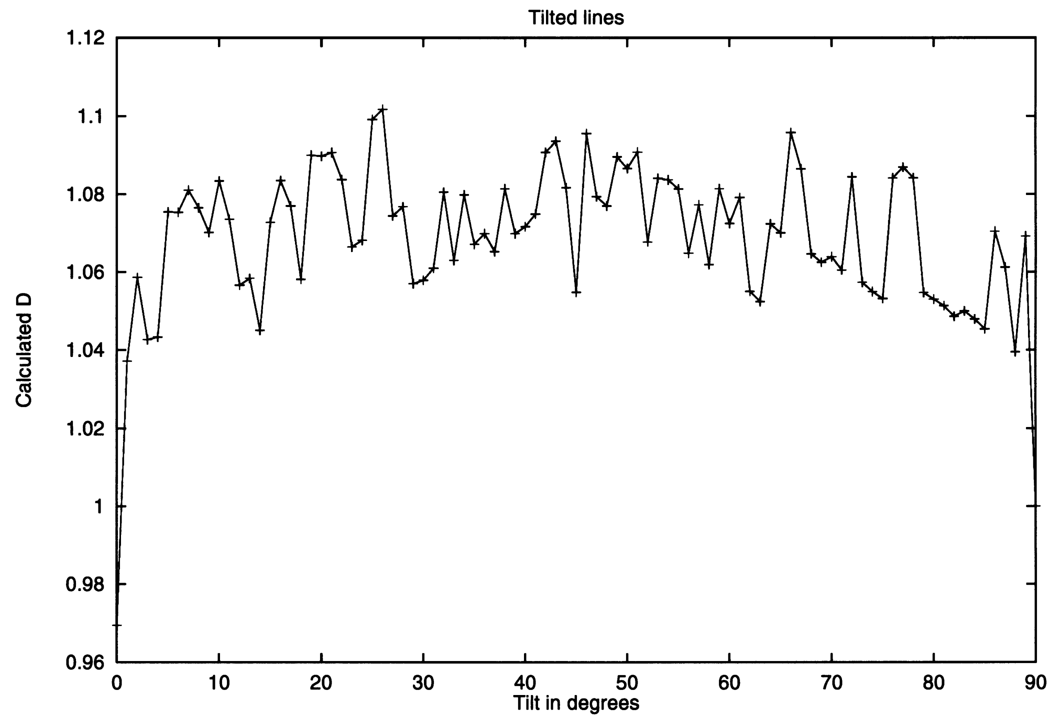

Figure 7. Calculated *D* for a tilted 1024-pixel-long line.

3.5 Errors caused by scanner resolution

Up to this point, we have been working with computer-generated shapes. Compared with digitized actual images, these are unrealistic, because (1) they do not contain noise (that is, spurious pixels), (2) they have exactly the resolution we choose (namely, a power of 2), and, more importantly, (3) they were all made by 1-pixel-thick lines. Real images digitized by a scanner will never exhibit such optimal characteristics. In fact, the scanner can introduce unexpected detrimental effects in the image quality.

Leaving the problem of image retouching aside (several software packages are available to this end), let us concentrate on the scanner resolution. Nearly all scanners can digitize images

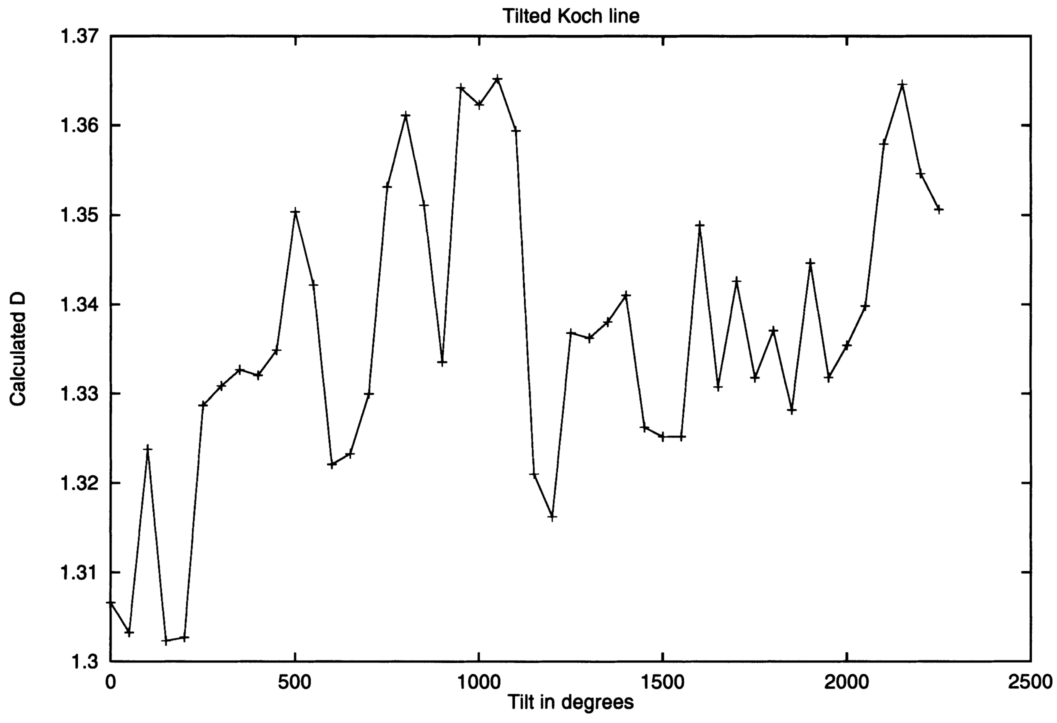

Figure 8. Calculated *D* for a tilted 1024-pixel-long Koch curve.

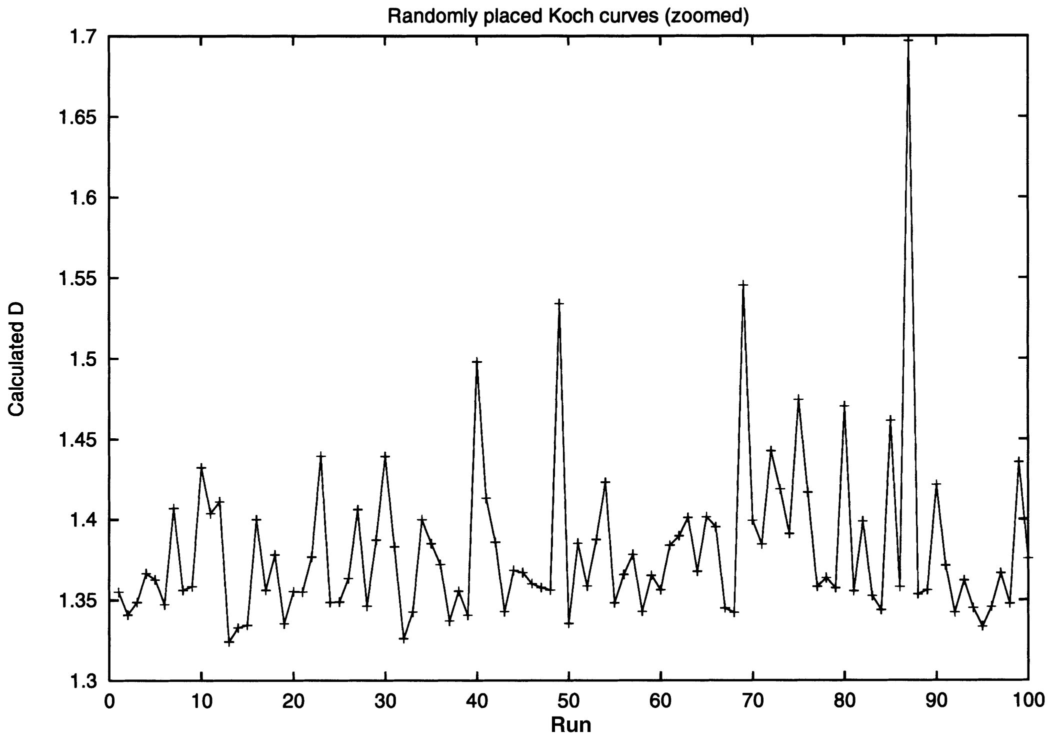

Figure 9. 100 runs of a simulation involving the placement of a Koch curve of random length, angle and position on a 4096-pixel-wide screen; average $D=1.38\pm0.05$.

at A4 paper size, that is, about 21×30 cm (8.27×11.81 inches). Since the image must be square, this leaves one with a usable 21×21 cm image size. At 600 DPI, a reasonable resolution for digitizing drawings, this gives a maximum box side of 4960 pixels; this is a good box bounding side for sensible analysis.

At this stage, the problem is how to make sure that the images are made of exactly 1-pixel-thick lines, which corresponds to 0.04 mm on paper. Thicker lines will be digitized as stripes, which have a theoretical D equal to 2, while thinner ones will not be digitized at all. In practice, the latter case is very unlikely with good-quality scanners; the thinner lines in good-quality maps are approximately 0.2–0.5 mm thick, and lines 1 mm thick are also common. At 600 DPI, any of the above lines will be digitized as a stripe with a width of the order of 10^1 pixels.

This is potentially a severe cause of bias. To assess the effect of thickness, we ran a simulation generating lines of thicknesses ranging from 1 to 100 pixels; the calculated D is shown in Fig. 10. As expected, the calculated fractal dimension is heavily biased. Repeating the experiment involving the placement of random lines and adding a random thickness from 1 to 50 pixels, the zoomed case exhibits an average bias of 38 per cent, peaking up to 50 per cent (Fig. 11).

All modern scanners, moreover, support resolutions of 1200 DPI or more. It must be noted that, paradoxically, the better the scanner resolution is, the worse the results of the fractal analysis will be. In fact, considering a 1-mm-thick line on paper, the digitized line thickness will be proportional to the scanner resolution, ranging from 3 pixels at 75 DPI to 94 pixels at 2400 DPI. To sum up, the scanner resolution is the most prominent source of bias in the digitized image.

There is a simple solution to overcome this apparent impasse: the map must be redrawn, making sure that all lines are 1 pixel thick. Needless to say, manually redrawing a map is a tedious process, which can become excruciating if the map is moderately to highly complex. However, there are computer programs (e.g. CorelTrace™) for turning bitmaps into collections of vectors and/or Bézier curves. Once the bitmap has been turned into vectors, these can be turned back into exactly 1-pixel-thick lines and/or curves. This method also allows one to resize the map to give it the desired size, optimally a power of 2.

3.6 Errors caused by the shape topology

Finally we considered the possibility that the image is composed of unconnected parts. There are two possible situations: (i) the image is inherently unconnected, or (ii) the map is actually a collection of unrelated parts. The first case is well represented by the Sierpinski gasket, whilst an instance of the second case is a tectonic map of a large area in the form of a digitized image of the mapped fault breaks.

It is difficult to tell by eye if an image shows clear evidence of unconnected self-similarity, but this will be apparent if the image is analysed as a whole. Applying the box-counting algorithm to this case quickly captures the fractal self-similar essence of the image. If such evidence is found, one should perform the box counting on unconnected parts. In fact, these parts do not necessarily belong to the same shape and must be analysed separately. A simple case of this situation is an image composed of two horizontal line segments separated by a gap. Note that this shape is completely different from the Cantor dust: the latter has a construction method that is recursively applied to each unconnected part. It is quite clear that performing the box counting on the whole image would yield a value of D smaller than 1, since the gap would cause several

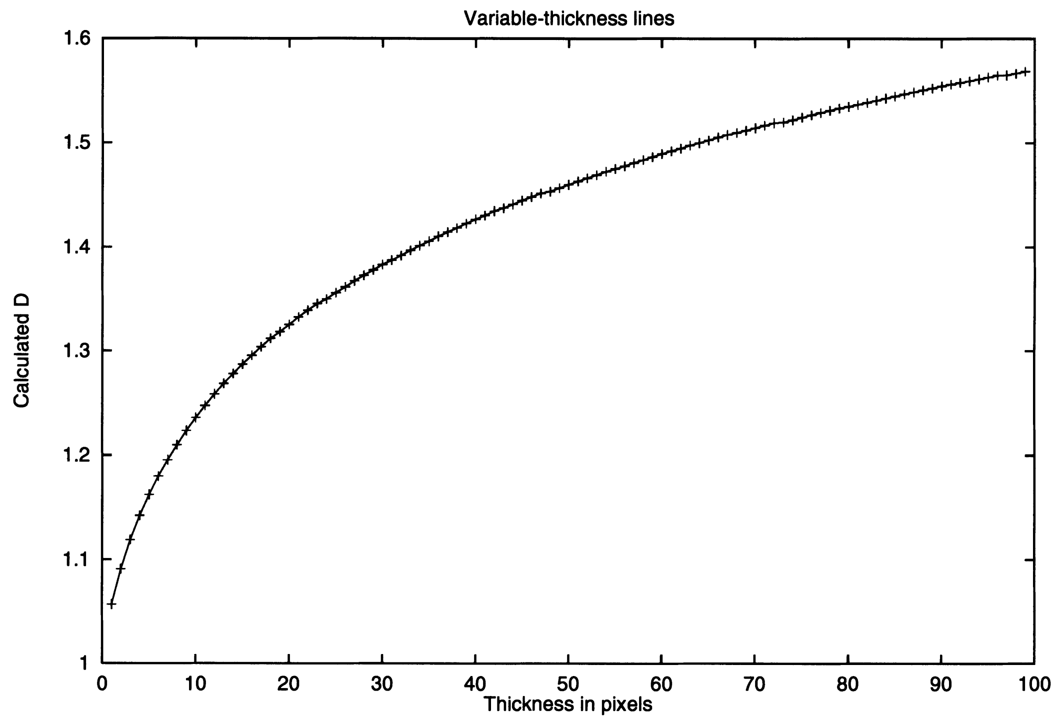

Figure 10. Calculated *D* for lines with thickness ranging from 1 to 100 pixels.

sub-boxes to be empty. On the other hand, no-one would call such an image a fractal: two separated images should be considered instead, each with dimension 1. This possibility was introduced into our code.

Considering a map representing an inherently unconnected fractal shape such as the Cantor dust or the Sierpinski gasket, extracting sub-parts would only separate clumps of pixels or, in the worst case, single pixels. The resulting analysis would not

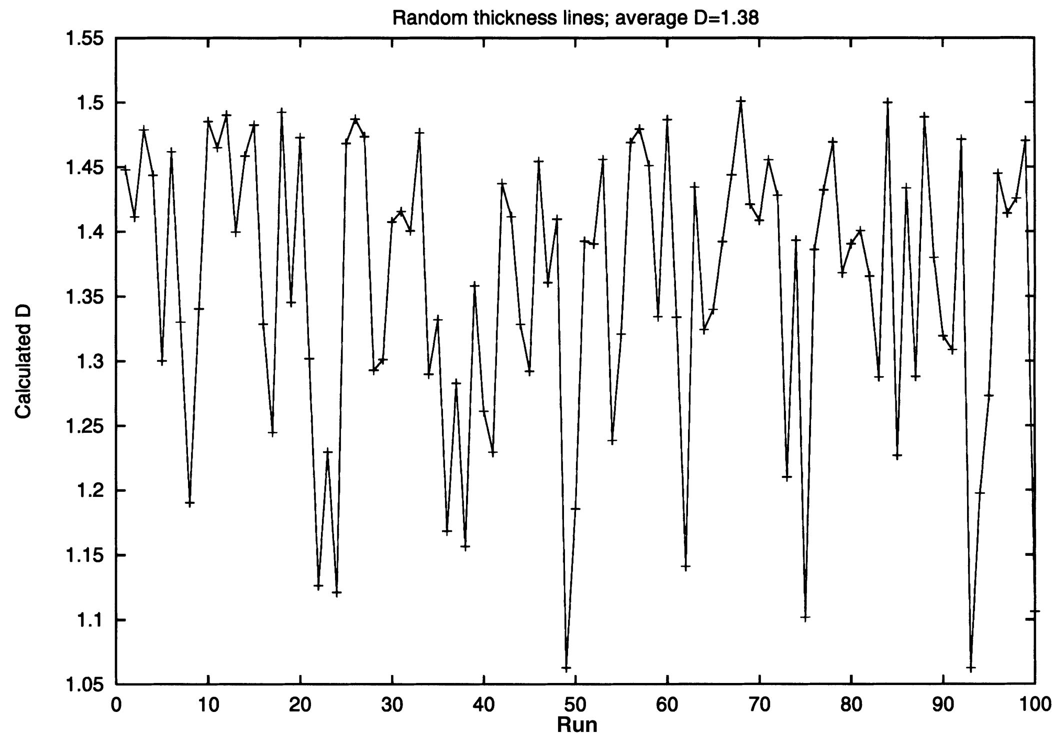

Figure 11. 100 runs of a simulation involving the placement of a line segment of random length (<2048 pixels), angle, position and thickness (<50 pixels) on a 4096-pixel-wide screen; average $D = 1.38 \pm 0.12$.

have any meaning at all. However, their fractal nature would readily emerge from the analysis of the image as a whole, which is always the first operation to be performed.

4 PERFORMING A CORRECT ANALYSIS

With all these problems in mind, let us define how a box-counting analysis should be conducted in practice, assuming that a high-quality map is used. To begin with, we shall not proceed manually, always using computer software instead.

Once the map has been digitized, one applies a vectorization algorithm to the map to ensure 1 pixel thickness of curves. An optimal image side can also be specified. Next, one examines the existence of preferential directions; if the map shows that there are such directions (e.g. a tectonic lineament), the bounding box should be chosen to be parallel to them to minimize the bias introduced by the angle.

The image is then ready to be analysed as a whole for self-similarity through box counting. If a positive result is obtained, the analysis may end. If no positive result is obtained, the map must be divided into its possible unconnected parts, each of which is to be analysed individually using a program with zooming capability. In all cases, the box-counting analysis must compensate for angle bias by taking at least 20 random placements and averaging the values of D obtained. Finally, since even when taking all these precautions, measurements of D positively biased by 10–20 per cent are to expected, the final estimate should be multiplied by 0.8–0.9 to provide the best possible estimate.

5 CONCLUSIONS

We have shown that several sources of bias affect the process of scanning and analysing an image, and that the resulting fractal dimension can be seriously affected. Some bias results from the incorrect framing of the image and can be removed, or at least minimized; other bias is inherent in the digitization process and cannot be avoided. As a consequence, obtaining the exact value of D is never possible; a residual bias of 10 per cent should always be expected.

The procedure to be followed in order to obtain the best results has been outlined. Moreover, computer software has been developed to bypass most of the sources of error, and to automate the process of digitizing and scanning the image. This program is available as freeware at our web site (see Appendix A).

REFERENCES

Gonzato, G., 1998. A practical implementation of the box counting algorithm, *Comput. Geosci.,* **24,** 95–100.

Gonzato, G., Mulargia, F. & Marzocchi, W., 1998. Practical application of fractal analysis: problems and solutions, *Geophys. J. Int.,* **132,** 275–282.

Grassberger, P., 1993. On efficient box counting algorithms, *Int. J. Mod. Phys. C,* **4,** 515–523.

Hamburger, D., Biham, O. & Avnir, D., 1996. Apparent fractality emerging from models of random distributions, *Phys. Rev. E,* **53,** 3342–3358.

Malcai, O., Lidar, D.A., Biham, O. & Avnir, D., 1997. Scaling range and cutoffs in empirical fractals, *Phys. Rev. E,* **56,** 2817–2828.

Mandelbrot, B.B., 1967. How long is the coast of Britain? Statistical self-similarity and fractional dimension, *Science,* **156,** 636–638.

Mandelbrot, B.B., 1982. *The Fractal Geometry of Nature,* W. H. Freeman, New York.

Ouillon, G. & Sornette, D., 1996. Unbiased multifractal analysis: application to fault patterns, *Geophys. Res. Lett.,* **23,** 3409–3412.

Ouillon, G., Castaing, C. & Sornette, D., 1996. Hierarchical scaling of faulting, *J. geophys. Res.,* **101,** 5477–5487.

Turcotte, D.L., 1989. Fractals in geology and geophysics, *Pageoph,* **131,** 171–196.

APPENDIX A: COMPUTER CODE

The code VSBC2 is essentially identical to the original version of VSBC (Gonzato 1998), but features a zooming capability, that is, a routine for loading maps saved in the common Windows .bmp format, and the capability of extracting all the unconnected parts from the map. The latter routine is massively recursive; this means that systems with small amounts of memory might not be able to run VSBC2 unless the option `--whole`, which disables the extraction of subimages, is specified. We were unable to find a computer with less than 8 MByte RAM to test this possibility.

The program has been compiled and tested with standard PC hardware under Linux 2.0.36 (using the C compiler gcc, version 2.7.2.3) and Windows 95 (using the Mingw32 port of gcc), as well as with an Alpha Server under Digital Unix 4.0. Being written in standard ANSI C, VSBC2 should compile with no modification under other systems that provide an ANSI C compiler and the utility `make`. Compilation instructions are included in the archive.

The program source is available from ftp://ibogeo.df.unibo.it/ (137.204.48.138), in directory/pub/vsbc.