Computers & Geosciences 37 (2011) 241–249

Contents lists available at ScienceDirect

Computers & Geosciences

journal homepage: www.elsevier.com/locate/cageo

A fast MATLAB program to estimate the multifractal spectrum of multidimensional data: Application to fractures

Jeferson de Souza *, Sidnei Pires Rostirolla [1]

Universidade Federal do Paraná, Centro Politécnico, Laboratório de Análise Bacias & Petrofísica, Caixa Postal 19027, CEP 81531-990 Curitiba, PR, Brazil

ARTICLE INFO

Article history:
Received 1 November 2007
Received in revised form
9 September 2010
Accepted 25 September 2010
Available online 9 November 2010

Keywords:
Fractal
Multifractal
Fractures
Box-counting
MATLAB
Algorithm

ABSTRACT

A MATLAB® program based on the Hou algorithm for estimation of fractal dimension and multifractal spectrum of fractures is presented. The program performance was tested with many synthetical fractals and field data. Interpolation and sampling effects on the fractal dimension and multifractal spectrum estimation were also studied. Some common problems related to the fractal dimension and multifractal spectrum are also discussed.

© 2010 Elsevier Ltd. All rights reserved.

1. Introduction

Fractal and multifractal-based methods have been successfully applied in many fields of geosciences (Agterberg and Cheng, 1999). They can provide valuable information on the statistical and geometrical properties of geological and geophysical variables.

The concept of fractals was introduced by Mandelbrot (1983) to describe objects whose properties have a power-law dependence on the scale. Fractal geometry is scale-invariant, i.e. the set in one given scale is similar to the set viewed in another scale (self-similarity). We would like to remind the reader that power-law does not imply self-similarity or fractality (Bour et al., 2002; Bour and Davy, 1997) although some authors refer to power-law distributions as fractal distribution and unlike mathematical fractals, geophysical and geological phenomena present fractal behavior within a limited scale range.

There is a wide range of geological and geophysical phenomena that exhibit fractal behavior. Examples include coastline perimeter (Mandelbrot, 1983), frequency–intensity distribution of earthquakes (Gutenberg–Richter Law) (Turcotte, 1992), oil and gas field distribution (Hein, 1999), earth gravity field (Thorarinsson and Magnusson, 1990), geomagnetic reverse records (Cortini and Barton, 1994), among others.

Spatial variation in physical properties of geological media that control migration, trapping and flow of hydrocarbon, e.g., rock porosity, hydraulic conductivity, gouge particle, fracture spatial distribution (Yielding et al., 1996), can present fractal characteristics. For this reason software for analysis and modeling of fractured rock masses have incorporated fractal based models in its set of tools (Dershowitz et al., 1998).

In particular, in recent years, spatial fracture distribution has been described in terms of fractal geometry and several works have shown its relevance in Structural Geology (Pérez-López and Paredes, 2006) and Petroleum Geology (La Pointe and Barton, 1995). Geometrical features of fracture are important to predict the fracturing bellow seismic scale (Gauthier and Lake, 1993), to predict the number of unobserved fractures from a sample of a restrict part of fracture population (Yielding et al., 1996), quantification of fracture anisotropy (Pérez-López and Paredes, 2006), reservoir characterization (Odling et al., 1999), fluid flow in reservoirs (Odling et al., 1999), flow and transport in porous media and fractured rock (Sahimi, 1993), simulation of fracture networks for reservoirs (Tran et al., 2005).

Box-counting method has been largely used to estimate fractal dimensions of measures, but this procedure has been pointed out as problematic due to memory and time limitations (Hou et al., 1990). Despite the fact that the method has been object of criticism, especially when applied to real stochastic data (Sornette et al., 1993; Berkowitz and Hadad, 1997; Cowie et al., 1993; Ouillon and Sornette, 1996; Ouillon et al., 1996), recent works have indicated that box-counting based methods are realistic for estimating fractal dimensions from real a synthetical data (Roy et al., 2007; Verbovsek, 2009). Furthermore,

* Corresponding author.
E-mail addresses: jdesouza@ufpr.br (J. de Souza), sidnei.rostirolla@vale-ep.com (S. Pires Rostirolla).
[1] Present address: Vale E & P, Av. Graça Aranha, 26, 4 º andar, Rio de Janeiro, RJ, Brazil.

0098-3004/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cageo.2010.09.001

many problems in calculation of fractal dimension, as for instance finite-size effects, have been reported in literature (Gonzato et al., 1998, 2000; Bonnet et al., 2001). Thus, different box-counting based methods for estimating fractal dimensions have been proposed in order to improve the accuracy of estimation and decrease computation time. For example, Agterberg et al. (1996) proposed a procedure to avoid bias due to both the lack of exposure and edge effects. One of the most efficient box-counting algorithms has been proposed by Hou et al. (1990). The Hou algorithm has computational complexity of $O(N)$, where N is the number of points, differently from most box-counting algorithms which have complexity of $O(N^{D_E})$, where D_E is the dimension of Euclidean space where the fractal is embedded.

Recent works make use of large data sets and/or several data sets. In particular, the estimation of the fractal dimension of fracture by using box-counting based methods is very difficult because interpolation of points along individual fractures increases severely the number of data points. In order to make feasible such works, one needs to use efficient and fast algorithms.

In this work, an efficient MATLAB program for fast computation of fractal dimension and multifractal spectrum of fractures is presented. To test the performance of the program we use synthetic fractals and field data. The effects of different sampling rates and interpolation intervals on the fractal dimension estimation are also studied with the program.

2. Theory

The term fractal is used to describe geometrical objects or functions which are scale invariant, i.e. the part of object (or function) is similar to the whole (self-similarity) (Feder, 1988). Sets whose properties have fractal geometry have a power-law dependence on the scale and the power is the so-called fractal (or Hausdorff) dimension D_f. Consider a set which is embedding in a (hyper)-volume in E-dimensional Euclidean space with maximal linear length L. The fractal dimension is usually calculated by covering the object with (hyper)-boxes of linear length $\varepsilon \le L$ and (hyper)-volume given by ε^{D_E}, where $D_E \ge D_f$ is the dimension of Euclidean space where the fractal is embedded, and counting the number $N_{box}(\varepsilon)$ of boxes that contain points:

$$N_{box}(\varepsilon) \sim \varepsilon_f^D. \tag{1}$$

The fractal dimension is obtained by evaluating

$$D_f = \lim_{\varepsilon \to \infty} \frac{\log N_{box}(\varepsilon)}{\log \varepsilon}. \tag{2}$$

Multifractals are related to the statistical distribution of measures on a geometrical support: a line, a surface, a volume, or a fractal, for instance (Feder, 1988). Multifractals are formed by an interwoven of fractal subsets with different scaling exponents. Once again let us consider a set which is embedding in a (hyper)-volume in E-dimensional Euclidean space with maximal linear length L. Let us cover the set with (hyper)-boxes of linear length $\varepsilon \le L$ and (hyper)-volume ε^{D_E}. The normalized probabilities for measure (or mass) in the i-th box, in the resolution ε, is given by $N_i(\varepsilon \le L)/N$, where N is the total number of points and N_i the number of points in the i-th box:

$$P_i(\varepsilon) = \frac{N_i(\varepsilon)}{N}. \tag{3}$$

For multifractal measures, for the i-th box in scale ε, P_i scales as

$$P_i(\varepsilon) \sim \varepsilon^{\alpha_i}, \tag{4}$$

where α_i is the local Lipschitz–Hölder exponent. The number of boxes in which $P_i(\varepsilon)$ has Lipschitz–Hölder exponent equals α is given by

$$N_\alpha(\varepsilon) \sim \varepsilon^{f(\alpha)}. \tag{5}$$

Thus, for a given α, as $\varepsilon \to 0$, $f(\alpha)$ provide the dimension of the subset of the measure that has Lipschitz–Hölder α. To obtain the $f(\alpha)$, it is suitable to define a partition function:

$$Z_q(\varepsilon) = \sum_i^{n_i(\varepsilon)} P_i^q, \tag{6}$$

where $n_i(\varepsilon)$ is the number of boxes in the scale ε and the q parameter ($q \in \Re$) is called *moment* of order q. If the measure has a multifractal distribution, $Z_q(\varepsilon)$ scale with ε in the following way:

$$Z_q(\varepsilon) \sim \varepsilon^{\tau_q}, \tag{7}$$

where τ is the correlation exponent or mass exponent of order q. If we know the partition function (6), we can determine the $f(\alpha)$ function, by evaluating $\tau(q)$:

$$\tau(q) = \lim_{\varepsilon \to 0} \frac{\log Z_q(\varepsilon)}{\log \varepsilon}. \tag{8}$$

The variables f and α are related to $\tau(q)$ via Legendre transform:

$$f(\alpha_q) = q\alpha_q - \tau_q, \tag{9}$$

$$\alpha_q = \frac{d\tau_q}{dq}. \tag{10}$$

The $f(\alpha)$ function provide the fractal dimension f of the sub-set of the measure which has a Lipschitz–Hölder exponent α.

The $f(\alpha)$ spectrum (or multifractal spectrum) is related to the *generalized dimensions* D_q (Halsey et al., 1986):

$$D_q = \frac{1}{q-1}[q\alpha(q) - f(\alpha(q))]. \tag{11}$$

When $q=0$, $f(\alpha_{q=0}) = \max[f(\alpha)]$ represents the fractal dimension of the support of the measure D_0. For $q = \pm\infty$, one has D_∞ and $D_{-\infty}$, corresponding to the values α_{max} and α_{min}, respectively. In theory, $\Delta\alpha = \alpha_{max} - \alpha_{min}$ should be zero for monofractals (fractals characterized by a single fractal dimension). However, numerically estimated $f(\alpha)$ spectra for monofractals present a narrow but non-zero $\Delta\alpha$, due to finite size effects.

3. Algorithm

The Hou et al. (1990) algorithm is an improvement of the algorithm proposed by Liebovitch and Toth (1989). It is based on the fact that the coordinates of the fractal—suitably shifted and rescaled—written in binary numerical base can be combined to form bits strings with $k \cdot D_E$ bits and whose first $m \cdot D_E$ bits from left right determine uniquely the position of the coordinates in D_E-dimensional space. Here, $m = 1,2,\ldots,k$ and k is a positive integer. Thus, the $N \cdot D_E$ coordinates are mapped in N bit strings with $k \cdot D_E$ bits, where k is the maximal number of bits used to represent each coordinate in binary base. After masking $m \cdot D_E$ bits from right to left, strings that have the same position code belong to the same box in resolution m. A stand-alone box-counting program based on Hou algorithm for fractal dimension estimation was presented by Kruger (1996). Hou algorithm is composed of five steps:

(1) The original data are mapped into interval $(0, 2^k - 1)$. Thus, the new shifted and normalized coordinates can be stored in a unsigned integer vector.
(2) The new data are then converted from decimal to binary numerical base and each coordinate have k bits. For example, for a set embedded in a two-dimensional space, one has $(x)_{\text{dec}} = (x_k x_{k-1} \cdots x_2 x_1)_{\text{bin}}$ and $(y)_{\text{dec}} = (y_k y_{k-1} \cdots y_2 y_1)_{\text{bin}}$, where the subindex "dec" and "bin" represent the coordinates in decimal and binary numerical base, respectively and x_m and y_m can assume the values 0 or 1.

Table 1
Columns 1,2 and 3,4 show *xy* coordinates on decimal and binary base, respectively. Column 5 shows bits of columns 3 and 4 intercalated (position codes) and column 6 shows position codes sorted. In column 7 are shown masked bits. Remark that points belonging to same box have same position code. Columns 8,9, show the same procedures of columns 5–7, but with bits mapped into unsigned integers.

x_{dec}	y_{dec}	x_{bin}	y_{bin}	interc.	sort	mask	interc.	sort	mask
0	0	00	00	0000	0000	00	0 0	0 0	0
0	1	00	01	0010	0001	00	0 2	0 1	0
0	2	00	10	1000	0010	00	2 0	0 2	0
0	3	00	11	1010	0011	00	2 2	0 3	0
1	0	01	00	0001	0100	01	0 1	1 0	1
1	1	01	01	0011	0101	01	0 3	1 1	1
1	2	01	10	1001	0110	01	2 1	1 2	1
1	3	01	11	1011	0111	01	2 3	1 3	1
2	0	10	00	0100	1000	10	1 0	2 0	2
2	1	10	01	0110	1001	10	1 2	2 1	2
2	2	10	10	1100	1010	10	3 0	2 2	2
2	3	10	11	1110	1011	10	3 2	2 3	2
3	0	11	00	0101	1100	11	1 1	3 0	3
3	1	11	01	0111	1101	11	1 3	3 1	3
3	2	11	10	1101	1110	11	3 1	3 2	3
3	3	11	11	1111	1111	11	3 3	3 3	3

(3) For each one of N data points of the set, a bit string by intercalating the bits of D_E coordinates is constructed. For $D_E=2$, one has

$$x_{k-1}y_{k-1}x_{k-2}y_{k-2}\cdots x_1y_1x_0y_0.$$

This string is referred as a *position code* for the data point, since it determine uniquely the position of coordinate in E-dimensional space. The D_E bits belonging to the m-th position can be mapped into a single unsigned integer number in the interval $(1,2^{D_E}-1)$. For $D_E=2$ the bits sequences 00, 01, 10 and 11 are converted to 0,1,2 and 3, respectively. This procedure require less memory because for this case one needs to allocate 8 bits for the k-th position, instead $D_E \cdot 8$ when the bit strings is in binary base.

(4) The new coordinate (position code) are sorted in lexicographical order.

(5) The $m \cdot D_E$ first bits from right to left are masked. Points that belong to the same box have the same masked bit string. The masking procedure is repeated k times and in each iteration D_E more bits are masked:

$$\begin{aligned}
&x_{k-1}y_{k-1}x_{k-2}y_{k-2}\cdots x_1y_1\\
&x_{k-1}y_{k-1}x_{k-2}y_{k-2}\cdots x_2y_2\\
&\vdots\\
&x_{k-1}y_{k-1}x_{k-2}y_{k-2}\\
&x_{k-1}y_{k-1}
\end{aligned}$$

The five steps are summarized in Table 1.

Then, by scanning the N bit strings, the number of changes (monofractal) or the number of strings in each class of equal strings (multifractal case) is stored. For monofractal case the number of changes represents the number of box needed to cover the fractal set in the scale m. For multifractal case, in each scale, the number of equal strings represents the number of points contained within a single box. Thus, if one has l classes of strings in the resolution ε, each class will have $N_i(\varepsilon)$ strings (points) and $N_i(\varepsilon)/N$ represents the normalized probability for i-th box.

4. The box_count program

Box_count program has four functions. Function **data_prep** map the data into interval $(0,2^k-1)$. Function **bit_int** convert the vector from decimal to binary numerical base, intercalate the bits and convert the bits in m-th position to a unsigned integer number. To sort the matrix generated by **bit_int** function we use the MATLAB internal function sortrows. Function **bit_mask** masks the bits and calculates the number of non-empty box for monofractal option or call the **part_func** function for the multifractal option. Function **part_func** calculates the probabilities P_i (Eq. (3)) and the partition function (Eq. (6)). **bit_mask**, sortrows and **part_func** functions depend on the number of points only. They do not depend on the Euclidean dimension in which the set is embedded.

The input of the **box_count program** are: (1) the name of the *ASCII* file containing the data to be processed which may be a $N \times D_E$ or $D_E \times N$ vector, where is D_E embedding dimension and N is the number of points of the set, being $N > D_E$; (2) the number of bits (k) for representing the string for each coordinate, from 4 to 64 bits—for example if user choose $k=8$, coordinate will be rescaled to the interval $(0,2^8-1)$ and (3) the kind of analysis—or multifractal. After running the **box_count** program, users can use the auxiliary program **fit_frac** and **leg_transf** to get the fractal dimension and the $f(\alpha)$ curve, respectively. For the auxiliary programs, users may provide the binary logarithm of the lower and the upper cutoffs, m_{low} and m_{up}, which range from 1 to k.

By its very nature, the algorithm generates bins whose linear size increases following a power of two (logarithmic binning). Consequently the real size of the bins r is obtained by $r=(2^m/2^k)s$, where s is the maximal linear length of the set. Thus, in a $\log_2-\log_2$ plot the points are evenly spaced and the numbers on the abscissa are the values of m.

The program **fit_frac** generates a plot where upper and bottom axes correspond to the normalized and real scales, respectively. The dashed lines and the left axis correspond to number of box containing points and the dotted lines and the right axis correspond to the local fractal dimension. The plot exhibits the local fractal dimension on the right axis, in order to aid users to evaluate the best interval for which the scale regime holds. The program also provides the error for the fractal exponent, estimated by least-squares, but it does not contain the errors in data points, as discussed in Bonnet et al. (2001). The local fractal dimension is estimated by the numerical derivative of the number of box containing points $N_{box}(r)$, at the point r.

To run the program users may type in MATLAB prompt [**np, s, q**]=**box_count;**. After running the **box_count** program, users may type [**x y**]=**fit_frac**(**np**,**s**) for monofractal option or [**x y**]=**leg_transf**(**np**,**s**,**q**) for multifractal option. The real values of the lower and upper cutoffs r_{min} and r_{max}, the fractal dimension or the values of α and $f(\alpha)$ are printed in the prompt.

The program was tested on an IBM® computer with 2 Intel® processors Xeon(TM) 3.2 GHz and 4096 K RAM, running under Linux environment. The same code can also run in GNU Octave language, but in this case the program is not so fast, mainly because sortrows command is slower under Octave.

5. Synthetical examples

In this section we assess the accuracy and efficiency of our code by comparing its numerical results with those obtained from synthetical fractals and multifractals, for which the values of fractal dimension and the $f(\alpha)$ spectrum can be theoretically determined.

5.1. Monofractal

We have used well-known deterministic fractals, namely the Cantor set ($D_E=1$), Koch curve ($D_E=2$) and Sierpinski pyramid ($D_E=3$), in order to test the program performance. For these sets the

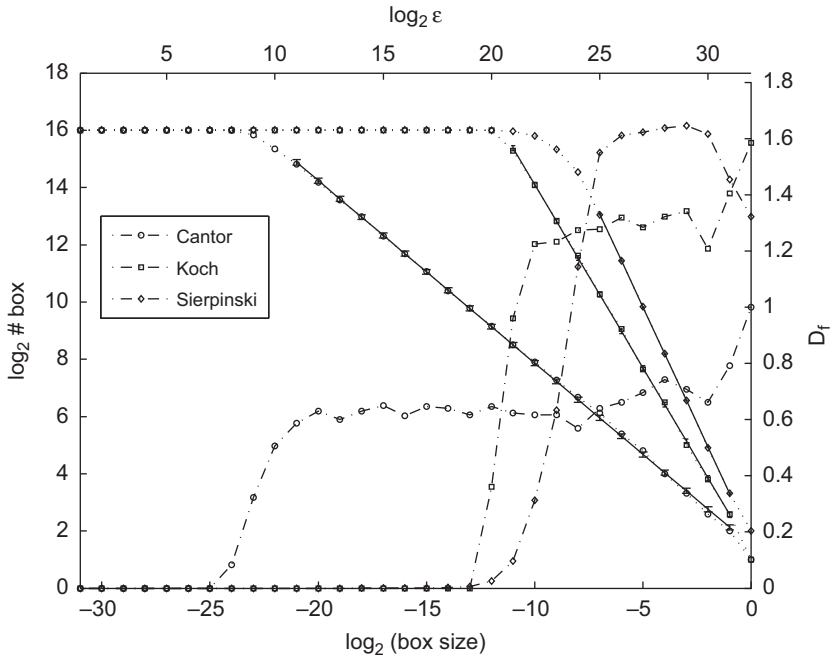

Fig. 1. Estimation of fractal dimension for Cantor set (open circles), Koch curve (squares) and Sierpinski Pyramid (diamonds), using 10^{16} points with $k=32$ bits. Solid lines are fitted curves. Upper and bottom axes correspond to the real and normalized scales, respectively. Dotted lines and left axis correspond to number of box containing points. Dashed lines and right axis correspond to the local fractal dimension.

fractal dimensions can be obtained analytically: $D_f = \log 2/\log 3 = 0.6309$, $D_f = \log 4/\log 3 = 1{,}2619$ and $D_f \log 4/\log 2 = 2$, respectively. Fig. 1 shows $\log_2(N) \times \log_2(r)$ plot for the Cantor set, Koch curve and Sierpinski pyramid, estimated with the **box_count** program from 2^{16} points and using strings of $k=32$ bits. Note that for a same number of points, the range—the interval limited by a upper and lower cutoff—increases as D_E increases. We generate *ASCII* files for each one of these fractals containing $N=2^{10}$, $N=2^{14}$ and $N=2^{20}$ points. The running times are shown in Table 2. We call $t_{\text{load}}(s)$ and $t_{\text{box-count}}(s)$ the times for loading the file to be processed into RAM disk and the running time for box-counting, respectively. Note that as the number of points decrease the estimation of fractal dimension became slightly inaccurate, specially for higher embedding dimension. Also, the higher embedding dimension, the smaller the interval which can be used to evaluate the fractal dimension.

To evaluate the program with different precision we have used the so-called Menger Sponge which is an example of D_E fractal, with $D_f = \log 20/\log 3 = 2.7268$. Running times for Menger sponge with 3.6×10^6 points for different k's are shown in Table 3. To test the program for Euclidean dimension higher than 3 we have use a 4-dimensional uniform distribution of points for which is expected a $D_f=4$. We have used 2^{20} points and we obtained $t_{\text{load}} = 15.80\,s$, $t_{\text{box-count}} = 20.65\,s$ and $D_f = 4.00$, for the interval $(m_{low}, m_{up}) = (29,31)$.

Table 2
Running times and fractal dimensions of some synthetical fractals with different numbers of points, estimated with **box_count** program using strings of $k=32$ bits. First column represents number of points. Second and third columns show times for loading files and running the program. Fourth and fifth columns show lower and upper cutoffs used to evaluate fractal dimension and last column shows estimated fractal dimensions.

Fractal	D_E	N	$t_{\text{load}}(s)$	$t_{\text{box-count}}(s)$	m_{low}	m_{up}	D_f
Cantor	1	2^{10}	0.005	0.004	19	31	0.64 ± 0.02
Cantor	1	2^{16}	0.25	0.34	10	31	0.64 ± 0.01
Cantor	1	2^{20}	4.06	6.33	7	31	0.63 ± 0.00
Koch	2	2^{10}	0.01	0.01	25	31	1.25 ± 0.06
Koch	2	2^{16}	0.50	0.84	21	31	1.28 ± 0.02
Koch	2	2^{20}	8.07	11.08	18	31	1.26 ± 0.01
Pyramid	3	2^{10}	0.01	0.01	29	31	1.88 ± 1.09
Pyramid	3	2^{16}	0.75	0.95	26	31	1.98 ± 0.10
Pyramid	3	2^{20}	12.01	13.01	24	31	1.99 ± 0.04

Table 3
Average running times and fractal dimension estimated using **box_count** for Menger sponge with 3.2×10^6, for different k's.

k	$t_{\text{box-count}}(s)$	m_{low}	m_{up}	D_f
8	16.49	2	7	2.73 ± 0.08
16	23.10	10	15	2.73 ± 0.08
32	38.32	26	31	2.73 ± 0.08

5.2. Multifractal

In order to evaluate the performance of our program for the multifractal case with $D_E=1$, we have used a the two-scale Cantor set for which is possible to construct a theoretical $f(\alpha)$ curve (Halsey et al., 1986). To generate a multifractal Cantor measure we use an *iterated function system* (IFS) (see Appendix A) and we have constructed a theoretical $f(\alpha)$ curve for this set by using the method described in Halsey et al. (1986). We have used stochastic IFS which generate multifractals closer to real data and are an alternative to the use of deterministic data with additional noise, as it has been pointed out by Ahammer and DeVaney (2005).

Once again, we have used data sets with three different sizes: $N=2^{10}$, 2^{14} and 2^{20}. Fig. 2 shows the $f(\alpha)$ curve for this three sets. Note that the multifractal spectrum is very sensitive to the lacking of points, mainly in the right part of $f(\alpha)$ curve where the partition function is weighted with negative moments ($q<0$, in Eq. (8)). For comparison, for each data set, we run the program using the fractal and multifractal options. The running times are shown in Table 4. For the two smaller sets most computation time is expended in

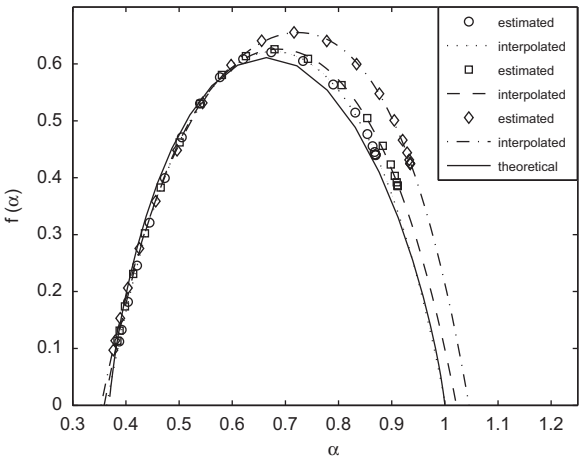

Fig. 2. Multifractal spectrum ($f(\alpha)$ function) for a two-scale Cantor set generated by IFS, estimated using **box-count** program with $k=32$ bits. We have used 10^{20} points with $(m_{low},m_{up})=(11,29)$ (open circles), 10^{14} points with $(m_{low},m_{up})=(18,30)$ (squares) and 10^{10} points with $(m_{low},m_{up})=(24,30)$ (diamonds). A theoretical $f(\alpha)$ curve (solid line) was plotted for comparison.

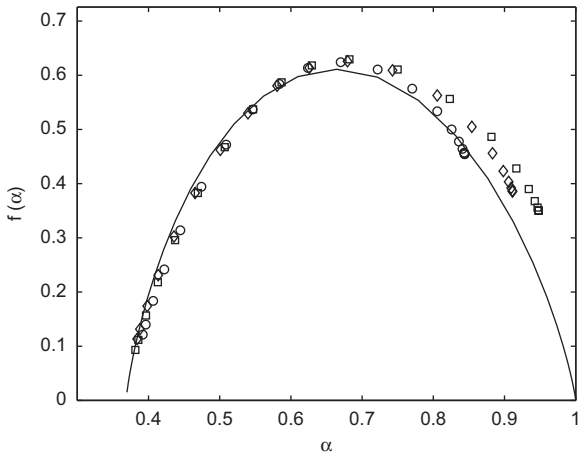

Fig. 3. Multifractal spectrum for a two-scale Cantor set with 10^{14} points fitted using different ranges: open circles—$(m_{low},m_{up})=(18,29)$; diamonds—$(m_{low},m_{up})=(18,30)$ and squares—$(m_{low},m_{up})=(19,30)$. Remark that, though $f(\alpha)$ spectrum is relatively robust to changes in values of cutoff for positive moments ($\alpha<\alpha(\max[f(\alpha)])$), curve is very sensitive for negative moments.

Table 4
Average running times for multifractal spectrum estimated using **box_count** with $k=32$, for two-scale Cantor set generate from IFS with different size.

Number of points	$t_{\text{box-count}}(s)$	$t_{\text{box-count}}(s)$
2^{10}	0.007	0.1
2^{14}	0.19	1.1
2^{20}	20	24

partition function estimation, but for the greater, running time for partition function is about $\frac{1}{5}$ of the total time.

We have studied the sensitivity of the $f(\alpha)$ spectrum to the choice of different ranges to fit the curves of Eq. (8). Fig. 3 shows, for the two-scale Cantor set with 10^{20} points and $k=32$ bits, four different $f(\alpha)$ curves for four different ranges. Note that though the left part of $f(\alpha)$ curve (positive q's) is robust to small changes in upper and lower cutoffs, the right part is very sensitive to them. Finally, the fractal dimension estimation for the two-scale Cantor set is slower than usual Cantor set with the same number of points. This happens because sorting procedure is slower for multifractal case.

To generate $D_E=2$ multifractals we use the IFS described in the website at Yale University sponsored by M. Frame, B. M. and N. Neger.[1] For this case it is possible obtain a theoretical $f(\alpha)$ curve. We have used the IFS described in Appendix A to generate two sets with support dimension $\max[f(\alpha)]=2$ (we call them set 1 and set 2). Set 1 has a broad spectrum of α values and set 2 has a narrow one. The average running times ($t_{\text{box-count}}$) for these IFS generated multifractals were, respectively, 21 and 14 s for fractal option and 30 and 54 s for multifractal option. We have used $k=32$ bits.Though the sets have the same size, the same embedding dimension and belong to the same family of multifractals, the computation times for set 1 is greater than set 2 for (mono)-fractal case and lesser for the multifractal case. Fig. 4 shows the estimated and theoretical $f(\alpha)$ curves for sets above mentioned. As it is expected, estimates of α_{max}'s are more inaccurate than α_{min}. Finally we stress that, due to finite-size effects, the algorithm is not able to evaluate points of $f(\alpha)$ curves for $q\to\pm\infty$, mainly for negative q's. Thus, the whole multifractal spectrum may be estimated by interpolation.

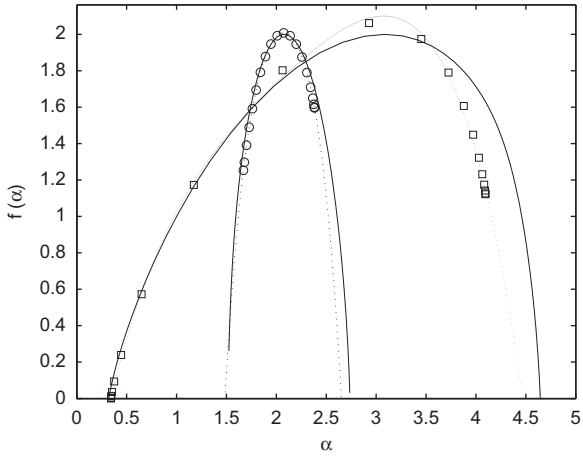

Fig. 4. Multifractal spectrum for two multifractal sets with 10^{20} points generated with IFS described in Appendix A. Open circles—estimated $f(\alpha)$ curve for set 1. Squares—estimated $f(\alpha)$ curve for set 2. Note that estimated curves are inaccurate for negative moments. Dotted lines are interpolated curves and solid lines are theoretical curves.

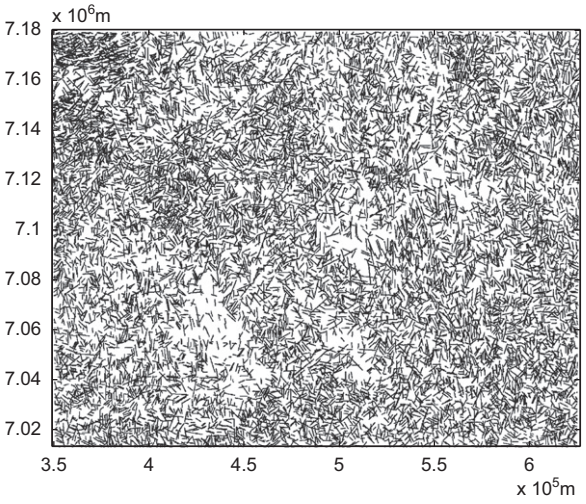

Fig. 5. Lineaments of area of study chosen for this work. This area is approximately 276 km in width and 166 km in length. Lineaments were interpreted from SRTM90 images (de Freitas et al., 2006).

[1] http://classes.yale.edu/fractals/MultiFractals/f(a)Curves/f(a)Curves.html

6. Field examples

The Paraná Basin is represented by a NS overall depression, classified as an intracratonic basin that extends into Brazil, Paraguay, Argentina and Uruguay. The studied area comprises partially the Paraná and Santa Catarina States in Southern Brazil, where multiple directions faults and tilting blocks are the main structural elements, which can be easily observed in remote sensing imagery. The quantified lineaments collected in SRTM images (see Fig. 5) exerts an important role in fluid migration along basin and were probably generated by reactivation of previous (pre-Cambrian) basement faults, among others formed during Phanerozoic basin evolution. The structural trends (de Freitas et al., 2006) corroborate the regional structural framework along the basin, whose analysis of brittle elements in images is suitable to test the functionality of the program.

Works which make use of either the barycenter or the fracture traces for fractal and multifractal modeling have been reported in literature (Bonnet et al., 2001). In this section we analyze these two cases.

The fractal dimension and the $f(\alpha)$ function for the centroids and the lineaments of the study area were estimated with the **box_count** program. Interpolation and sampling effects on the fractal dimension estimation was studied. We have also discussed the problem of the range of scale for which the fractal dimension can be defined.

To study the interpolation effects on the estimation of the fractal dimension, we generate data sets with different densities of points interpolated along the lineaments (between the ends of lineaments). The fractal dimension for the different interpolation intervals is shown in Table 5. We can see that there is a slight variation in fractal dimension for different interpolation interval. Fig. 6 shows the $\log_2(N) \times \log_2(r)$ plot for the three sets.

Comparison between the different interpolated data show that for very high interpolation interval the $\log_2(N) \times \log_2(r)$ plot exhibits two distinct regions with different slopes (besides finite-size plateau), in which one can define a straight line. This behavior is referred to in literature as "bifractal" (Volant and Grasso, 1996) and appears also in fracture study (Berkowitz and Hadad, 1997). The slope of the region between the finite size plateau and the last region is close to one—the dimension of a line. It is a indicative that it can be a spurious effect which occurs when the box size is of same order as the interpolation interval. In this interval of scales the algorithm is not detecting the fractal dimension of the spatial distribution of the fractures, but measuring the dimension of a line. Similar sampling effect has already been reported in literature (Walsh and Watterson, 1993; Bonnet et al., 2001; Bour et al., 2002).

Sampling effects are studied by resampling the lineaments in a lower sampling rate. Thus, we have created new sets with size $N/4$ and $N/8$, where $N=12\,576$, by randomly selecting some lineaments from original set. Such a procedure can mimic the effect of limited rock exposure, for example. This problem has already been addressed by Agterberg et al. (1996). Fig. 7 shows the effect of the undersampling on the fractal dimension estimation. Note that the fractal dimension is relatively robust to the undersampling effects but it decreases the interval which be can used to fit Eq. (2).

The same process was repeated for the estimation of $f(\alpha)$ function. Fig. 8 shows the $f(\alpha)$ function for the fractures centroids estimated using $k=20$ bits, for original and resampled data. It is apparent that resampled data do not reproduce the same spectrum

Table 5
Fractal dimension estimated from 12 576 lineaments with different interpolation intervals, using $k=16$ bits (Fig. 6). We have used the interval $(m_{low},m_{up})=(9,15)$ which correspond to lower and upper cutoffs $r_{min}=2159$ m and $r_{max}=138\,153$ m, respectively. Note that fractal dimension decreases as interpolation interval increase.

Density m^{-1}	# points	D_f	$t_{\text{load}}(s)$	$t_{\text{box-count}}(s)$
0.001	42 969	1.90 ± 0.07	0.34	0.41
0.01	373 663	1.92 ± 0.05	2.92	3.85
0.1	3 681 065	1.92 ± 0.05	28.77	39.03

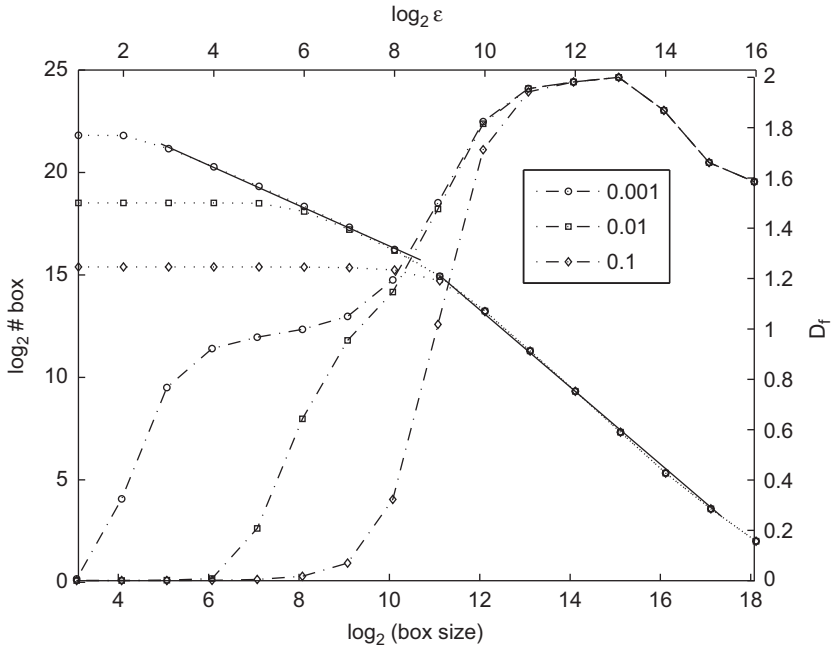

Fig. 6. Fractal dimension estimation of fractures for different interpolation intervals: 1 point/1000, 1 point/100 and 1 point/10 m. Not much information is added decreasing the interpolation interval. Furthermore, for a very high interpolation interval, a second region with different a slope appears in $\log(N) \times \log(r)$ plot. The slope is close to one (dimension of line), indicating that it is a spurious effect. Upper and bottom axes correspond to the real and normalized scales, respectively. Dotted lines and left axis correspond to number of box containing points. Dashed lines and right axis correspond to the local fractal dimension. Values of D_f are shown in Table 5.

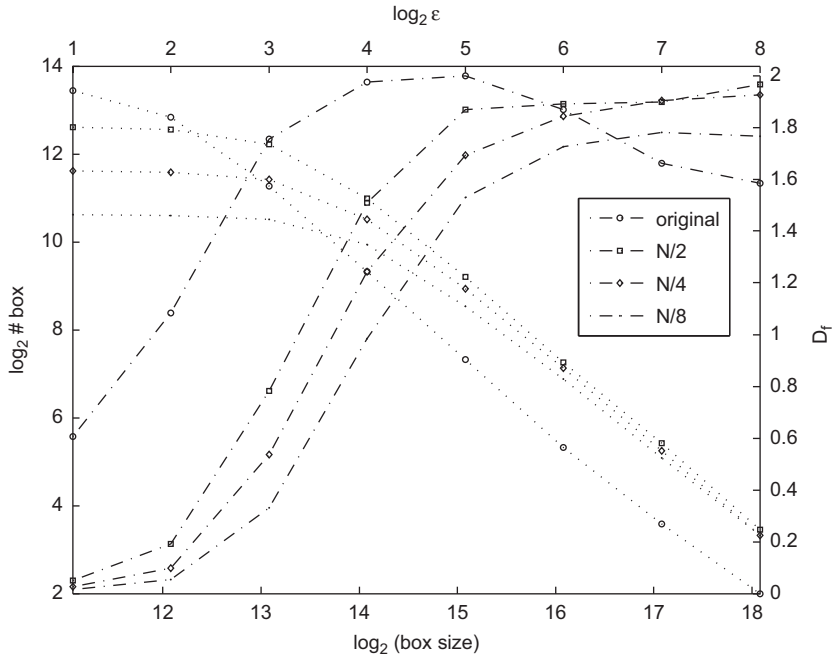

Fig. 7. Estimation of fractal dimension of fracture centroids for resampled data. Open circles—original data set with $N=12\,576$ centroids, $D_f=1.89$ for $(m_{low},m_{up})=(2,7)$. Squares—original region resampled by $N/2$ with $D_f=1.92$ for $(m_{low},m_{up})=(3,7)$. Diamonds—original region resampled by $N/4$ with $D_f=1.84$ for interval $(m_{low},m_{up})=(3,7)$. Full circles—original region resampled by $N/8$ with $D_f=1.87$ for interval $(m_{low},m_{up})=(4,7)$. Upper and bottom axes correspond to real and normalized scales, respectively. Dotted lines and left axis correspond to number of box containing points. Dashed lines and right axis correspond to local fractal dimension. Values of $m=2$, 3, 4 and 7 correspond to values of $r=$ 4327, 8654, 17 309 and 138 470 m, respectively.

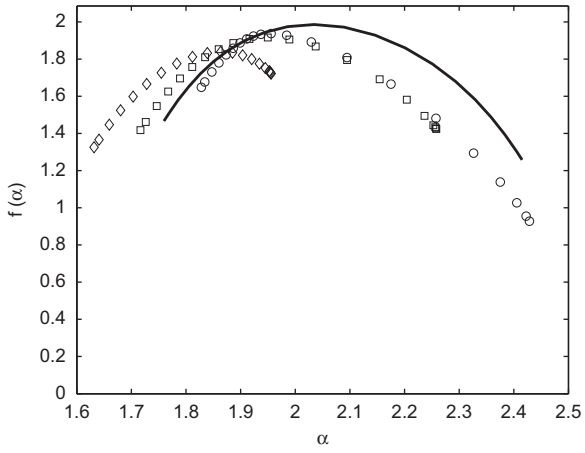

Fig. 8. Multifractal spectrum for fractures centroids of the area of study (open circles) and resampled sets with $N/2$ (squares) and $N/4$ (diamonds), estimated using **box-count** program with 32 bits precision and $(m_{low},m_{up})=(27,31)$ ($r_{min}=8634$ m and $r_{max}=138\,151$ m). For comparison we have also calculated multifractal spectrum of a two-dimensional uniform distribution using same parameters (solid line).

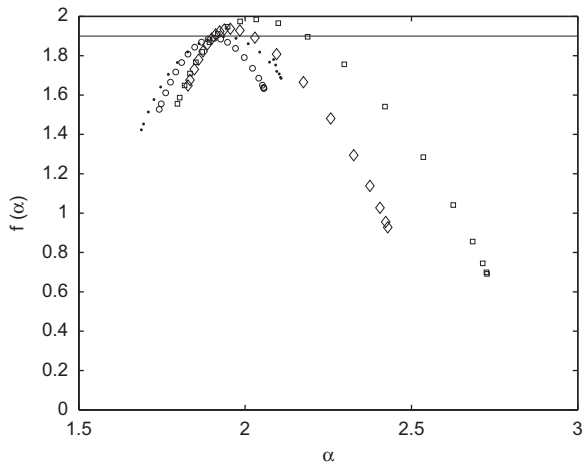

Fig. 9. $f(\alpha)$ function for fracture centroids estimated using $k=32$ bits and fitted using different ranges: $(m_{low},m_{up})=(26,30)$ (full circles), $(m_{low},m_{up})=(26,31)$ (open circles), $(m_{low},m_{up})=(27,30)$ (squares), $(m_{low},m_{up})=(27,31)$ (diamonds). Remark that choice of slightly different ranges affects shape of $f(\alpha)$ curve and/or the support dimension. Values of $m=26$, 27, 30 and 31 correspond to values of $r=$ 4317, 8634, 69 066 and 138 151 m, respectively.

as original data. Furthermore, it is evident that the range used to fit D_f strongly affect the shape of the $f(\alpha)$ curve (Fig. 9).

7. Discussion

We have presented a program for fast and accurate computation of the fractal dimension and $f(\alpha)$ spectrum of fractures. The performance of our program was evaluated by using different synthetical and real fractals, with different number of points. Simplicity of the program allow users to modify the program according to specific necessities. For example, Eq. (6) can be replaced by the expression proposed by Agterberg et al. (1996) for correction of bias due limited rock exposure.

We have studied sampling and interpolation effects using fields examples. For the fractures set studied in this work, a low density interpolation interval was enough to estimate the fractal dimension of fractures sets. Furthermore, for much denser interpolations a non-fractal interval in the $\log_2(N)\times\log_2(r)$ plot is introduced. This fact indicates that the behavior known as bifractality can be, in some cases, a spurious effect.

Therefore, for the region studied in this work, centroids can provide an unbiased measure of the spatial distribution of fracture,

as suggested by previous works (Bonnet et al., 2001; Bour et al., 2002; Darcel et al., 2003).

Sampling effects were evaluated by resampling the fractures at a low sampling rate. The fractal dimension estimation for under-sampled sets was not strongly affected, but the sampling effects severely affect the $f(\alpha)$ function computation. Hence, users are discouraged to work with multifractal analysis of small data sets.

As it has already been reported in literature (Bonnet et al., 2001), definition of lower and upper cutoffs is problematic and we have found that it is a serious problem specially for negative moments of the $f(\alpha)$ curve. For theoretical models one can use the support dimension, α_{max} and α_{min} to control the accuracy of the $f(\alpha)$ curve estimation, but in the case of field data only the support dimension can be found (by evaluating the (mono)-fractal dimension before the $f(\alpha)$ function) and one cannot rigorously define a $f(\alpha)$ curve. This fact is illustrated in Fig. 9: if is chosen the interval $(m_{low},m_{up})=(26,30)$ or $(m_{low},m_{up})=(26,31)$ for fitting Eq. (8), the support dimension will be approximately $\max[f(\alpha)]=1.9$ (close to fractal dimension of the set), but if the interval $(m_{low},m_{up})=(27,30)$ $(m_{low},m_{up})=(27,31)$ is chosen, one has $\max[f(\alpha)]=2$. Thus, we can disregard the last intervals, but we cannot define which of the first ones give the correct α_{max} and α_{min}.

The efficiency of the program can be improved if it is possible to estimate the threshold of finite size plateau. For higher embedding dimension, user are encouraged to use short strings (small k values), since the finite size plateau is larger for these cases.

We stress that *ASCII* files are obtained easily from most programs for lineaments interpretation.

Finally, besides geological applications, where embedding dimensions 1, 2 and 3 are common, the program can be used, for instance, for physical and dynamical systems applications, where common embedding dimensions are greater than 3.

Acknowledgments

The authors would like to thank S. Laroca and S.M.D. Queirós for reading of the paper, R.C. de Freitas for having supplied the geological data, M. Frame for comments and formulas for generating of theoretical $f(\alpha)$ curves and F. Mancini for his encouragement and support of this work. We would also like to thank the referees for their fruitful remarks and comments. The authors are also thankful to the Postgraduate Course in Geodetic Sciences of UFPR for softwares. This work was partially supported by CNPq and FINEP (Brazilian agencies) and PETROBRAS.

Appendix A. Generating multifractals from IFS

An iterated function system is a simple way to generate fractals and multifractals (Barnsley, 1933). The method consists in iterating two or more functions following certain probabilities. Different functions with different probabilities will generate different sets. For example, in the case of equation system (12)–(13), given a initial condition x_0, we randomly choose a number q ($q=0,1$ for example) and if $q=0$ we iterate Eq. (12), otherwise we iterate Eq. (13). Next, we randomly choose another q value and once again we iterate function (12) if $r=0$ or function (13) for $r=1$. Repeating the iteration N times one generate (multi)-fractals with N data points. In general, equal probabilities generate (mono)-fractals and unequal probabilities generate multifractals. Further details on the method can be found in Barnsley (1933) and at the website mentioned in Section 5.2. Here, we just describe the algorithms to generate the multifractals used to test the program. To generate a two-scale Cantor set we use the following algorithm: given an initial condition x_0, the consecutive values of x are obtained iterating the system of equations:

$$x(n+1)=1/2\cdot x(n) \quad \text{with probability } p1, \tag{12}$$

$$x(n+1)=0.4\cdot x(n)+0.6 \quad \text{with probability } p2, \tag{13}$$

where we have used $p1=0.25$ and $p2=0.4$ for generating the sets used to produce the curves of Fig. 2.

For generating a two-dimensional IFS we iterate with, probabilities p_i, one of the i system of equations below:

$$\begin{aligned} x(n+1)&=1/2\cdot x(n),\\ y(n+1)&=1/2\cdot y(n) \quad \text{with probability } p1, \end{aligned} \tag{14}$$

$$\begin{aligned} x(n+1)&=1/2\cdot x(n)+1/2,\\ y(n+1)&=1/2\cdot y(n) \quad \text{with probability } p2, \end{aligned} \tag{15}$$

$$\begin{aligned} x(n+1)&=1/2\cdot x(n),\\ y(n+1)&=1/2\cdot y(n)+1/2, \quad \text{with probability } p3, \end{aligned} \tag{16}$$

$$\begin{aligned} x(n+1)&=1/2\cdot x(n)+1/2,\\ y(n+1)&=1/2\cdot y(n)+1/2 \quad \text{with probability } p4, \end{aligned} \tag{17}$$

where we have used probabilities $(p1,p2,p3,p4)=(0.35,0.30,0.20,0.15)$ and $(p1,p2,p3,p4)=(0.80,0.10,0.06,0.04)$, for generating the two sets used to produce the $f(\alpha)$ curves in Fig. 4. The factor 1/2 are called *scaling factors* r_i. For these cases one has: $\alpha_{min}=\log(0.35)/\log(0.5)=1.51$, $\alpha_{max}=\log(0.15)/\log(0.5)=2.74$ and $\alpha_{min}=\log(0.80)/\log(0.5)=0.32$, $\alpha_{max}=\log(0.04)/\log(0.5)=4.64$, respectively. The theoretical $f(\alpha)$ curve for this can be obtained solving the *generalized Moran equation*: $p_1^q r_1^{\tau(q)}+\cdots+p_N^q r_N^{\tau(q)}=1$. For the cases mentioned above the scaling factors $r_i=r=1/2$ and $\tau(q)=\log(p_1^q+p_2^q+p_3^q+p_4^q)/\log(r)$. The $f(\alpha)$ spectrum can be obtained through Eqs. (9)–(10).

References

Agterberg, F.P., Cheng, Q., Brown, A., Good, D., 1996. Multifractal modeling of fractures in the Lac du Bonnet Batholith, Manitoba. Computers & Geosciences 22 (5), 497–507.

Agterberg, F.P., Cheng, Q., 1999. Introduction to special issue on fractals and multifractals. Computers & Geosciences 25 (9), 947–948.

Ahammer, H., DeVaney, T.T.J., 2005. The influence of noise on the generalized dimensions. Chaos, Solitons & Fractals 26 (3), 707–717 . doi:10.1016/j.chaos.2005.01.050.

Barnsley, M., 1933. Fractals Everywhere, second ed. Academic Press, Boston, MA 550pp.

Berkowitz, B., Hadad, A., 1997. Fractal and multifractal measures of natural and synthetic fracture networks. Journal of Geophysical Research 102 (B6), 12205–12218.

Bonnet, E., Bour, O., Odling, N.E., Davy, P., Main, I., Cowie, P., Berkowitz, B., 2001. Scaling of fracture systems in geological media. Reviews of Geophysics 39, 347–384 . doi:10.1029/1999RG000074.

Bour, O., Davy, P., 1997. Connectivity of random fault networks following a power law fault length distribution. Water Resources Research 33 (7), 1567–1583.

Bour, O., Davy, P., Darcel, C., Odling, N., 2002. A statistical scaling model for fracture network geometry, with validation on a multiscale mapping of a joint network (Hornelen Basin, Norway). Journal of Geophysical Research 107 (B6), 2113 . doi:10.1029/2001JB000176.

Cortini, M., Barton, C.C., 1994. Chaos in geomagnetic reversal records—a comparison between earths magnetic—field data and model disk dynamo data. Journal of Geophysical Research 99 (B9), 18021–18034.

Cowie, P.A., Vanneste, C., Sornette, D., 1993. Statistical physics model for the spatiotemporal evolution of faults. Journal of Geophysical Research 98 (B12), 21,809–21,821.

Darcel, C., Bour, O., Davy, P., de Dreuzy, J.R., 2003. Connectivity properties of two-dimensional fracture networks with stochastic fractal correlation. Water Resources Research 39 (10), 1272 . doi:10.1029/2002WR001628.

de Freitas, R.C., Rostirolla, S.P., Ferreira, F.J.F., 2006. Multithematic geoprocessing and structural analysis in the Irati Oil System—Rio Bonito, Paraná Basin, Petrobras. Geosciences Bulletin 14, 71–93.

Dershowitz, W., Lee, G., Geier, J., LaPointe, P.R., 1998. FracMan: Interactive Discrete Feature Data Analysis, Geometric Modelling and Exploration Simulation. User Documentation, Golder Associates Inc., Seattle, Washington.

Feder, J., 1988. Fractals. Plenum Press, New York 283pp.

Gauthier, B.D.M., Lake, S.D., 1993. Probabilistic modeling of faults below the limit of seismic resolution in Pelican field, North-Sea, Offshore United-Kingdom. AAPG Bulletin—American Association Of Petroleum Geologists 77, 761–777.

Gonzato, G., Mulargia, F., Marzocchi, W., 1998. Practical application of fractal analysis: problems and solutions. Geophysical Journal International 132 (2), 275–282 . doi:10.1046/j.1365-246x.1998.00461.x.

Gonzato, G., Mulargia, F., Ciccotti, M., 2000. Measuring the fractal dimensions of ideal and actual objects: implications for application in geology and geophysics. Geophysical Journal International 142, 108–116 . doi:10.1046/j.1365-246x.2000.00133.x.

Halsey, T.C., Jensen, M.H., Kadanoff, L.P., Procaccia, I., Shraiman, B.I., 1986. Fractal measures and their singularities—the characterization of strange sets. Physical Review A 33, 1141–1151 . doi:10.1103/PhysRevA.33.1141.

Hein, F.J., 1999. Mixed (“multi”) fractal analysis of granite wash fields/pools and structural lineaments, Peace River Arch area, northwestern Alberta, Canada; a potential approach for use in hydrocarbon exploration. Bulletin of Canadian Petroleum Geology 47, 556–572.

Hou, X.-J., Gilmore, R., Mindlin, G.B., Solari, H.G., 1990. An efficient algorithm for fast $O(N \ln(N))$ box counting. Physics Letters A 151, 43–46 . doi:10.1016/0375-9601(90)90844-E.

Kruger, A., 1996. Implementation of a fast box-counting algorithm. Computer Physics Communications 98, 224–234 . doi:10.1016/0010-4655(96)00080-X.

La Pointe, P.R., Barton, C.C., 1995. Creating reservoir simulations with fractal characteristics. In: Barton, C.C., La Pointe, P.R. (Eds.), Fractals in Petroleum Geology and Earth Processes. Plenum Press, New York, pp. 263–276.

Liebovitch, L.S., Toth, T., 1989. A fast algorithm to determine fractal dimensions by box-counting. Physics Letters A 141, 386–390 . doi:10.1016/0375-9601(89)90854-2.

Mandelbrot, B.B., 1983. The Fractal Geometry of Nature—Revised and Enlarged Edition. W.H. Freeman and Co, New York 495pp.

Odling, N.E., Gillespie, P., Bourgine, B., Castaing, C., Chilés, J.-P., Christensen, N.P., Fillion, E., Genter, A., Olsen, C., Thrane, L., Trice, R., Aarseth, E., Walsh, J.J., Watterson, J., 1999. Variations in fracture system geometry and their implications for fluid flow in fractured hydrocarbon reservoirs. Petroleum Geoscience 5, 373–384.

Ouillon, G., Castaing, C., Sornette, D., 1996. Hierarchical geometry of faulting. Journal of Geophysical Research 101 (B3), 5477–5487.

Ouillon, G., Sornette, D., 1996. Unbiased multifractal analysis: application to fault patterns. Geophysical Research Letters 23 (23), 3409–3412.

Pérez-López, R., Paredes, C., 2006. On measuring the fractal anisotropy of 2-D geometrical sets: application to the spatial distribution of fractures. Geoderma 134, 402–414 . doi:10.1016/j.geoderma.2006.03.022.

Roy, A., Perfect, E., Dunne, W.M., McKay, L.D., 2007. Fractal characterization of fracture networks: an improved box-counting technique. Journal of Geophysical Research 112, B12201 . doi:10.1029/2006JB004582.

Sahimi, M., 1993. Flow phenomena in rocks: from continuum models to fractals percolation, cellular automata and simulated annealing. Reviews in Modern Physics 65, 1393–1534 . doi:10.1103/RevModPhys.65.1393.

Sornette, A., Davy, P., Sornette, D., 1993. Fault growth in brittle–ductile experiments and the mechanics of continental collisions. Journal of Geophysical Research 98 (B7), 12111.

Thorarinsson, F., Magnusson, S.G., 1990. Bouguer density determination by fractal analysis. Geophysics 55, 932–935 . doi:10.1190/1.1442909.

Tran, N.H., Chen, Z.X., Rahman, S.S., 2005. Integrated conditional global optimization for discrete fracture network modelling. Computers & Geosciences 32, 17–27 . doi:10.1016/j.cageo.2005.03.019.

Turcotte, D.L., 1992. Fractals and Chaos in Geology and Geophysics. Cambridge University Press, Cambridge 221pp.

Yielding, G., Needham, T., Jones, H., 1996. Sampling of fault populations using subsurface data: a review. Journal of Structural Geology 18, 135–146 . doi:10.1016/S0191-8141(96)80039-3.

Verbovsek, T., 2009. TBCFD—a Visual Basic program for calculation of the fractal dimension of digitized geological image data using a box-counting technique. Geological Quarterly 53 (2), 241–248.

Volant, P., Grasso, J.-R., 1996. The finite extension of fractal geometry and power law distribution of shallow earthquakes: a geomechanical effect. Journal of Geophysical Research 99 (B11), 21879–21890.

Walsh, J.J., Watterson, J., 1993. Fractal analysis of fracture patterns using the standard box-counting technique: valid and invalid methodologies. Journal of Structural Geology 15 (12), 1509–1512.