An efficient, three-dimensional, anisotropic, fractional Brownian motion and truncated fractional Levy motion simulation algorithm based on successive random additions

Silong Lu\textsuperscript{a}, Fred J. Molz\textsuperscript{a,*,} Hui Hai Liu\textsuperscript{b}

\textsuperscript{a}Environmental Engineering and Science Department, Clemson University, 29625 Anderson, SC, USA
\textsuperscript{b}Earth Sciences Division, Lawrence-Berkeley National Laboratory, University of California, Berkeley, CA 94720, USA

Received 1 August 2001; received in revised form 10 July 2002; accepted 11 July 2002

Abstract

Fluid flow and solute transport in the subsurface are known to be strongly influenced by the heterogeneity of aquifers. To simulate aquifer properties, such as logarithmic hydraulic conductivity ($\ln(K)$) variations, fractional Brownian motion (fBm) and truncated fractional Levy motion (fLm) were suggested previously. In this paper, an efficient three-dimensional successive random additions (SRA) algorithm is presented to construct spatial $\ln(K)$ distributions. A convenient conditioning procedure using the inverse-distance-weighting method as a data interpolator, which forces the generated fBm or truncated fLm realization to go through known data points, is included also. The proposed method coded in the FORTRAN language, and a complementary code for verifying fractal structure in fBm realizations based on dispersional analysis, are validated carefully through numerical tests. These software packages allow one to go beyond the stationary stochastic process hydrology of the 1980s to the new geo-statistics of non-stationary stochastic processes with stationary increments, as embodied by the stochastic fractals fBm, fLm and their associated increments fGn and fLn.

Keywords: Fractional Brownian motion; Fractional levy motion; Anisotropy; Hydraulic conductivity; Conditioning

1. Introduction

The fact that heterogeneity of hydraulic conductivity ($K$) dominates fluid flow and solute transport in porous media has been known for many years. To simulate fluid flow and chemical transport in porous media in a more realistic manner, a relatively detailed, heterogeneous $K$ distribution is often needed. In practice, however, such detailed $K$ distributions usually are not available. Traditionally, various interpolation methods, such as several types of polynomial interpolation, inverse distance weighting, and Kriging, were widely used to estimate $K$ values between measurement points that are almost always sparse. In addition, the underlying property distributions were usually assumed to be represented by stationary stochastic processes (Gelhar, 1993). In contrast, the present work deals with non-stationary stochastic processes with stationary increments, an approach that is supported by data collected or analyzed over the past decade (Molz and Boman, 1993; Painter and Paterson, 1994; Painter, 1995, 1996a, b, 2001; Liu and Molz, 1997; Lu and Molz, 2001; Lu et al., 2002). The past methods based on...
stationary stochastic processes and various interpolation procedures tend to smooth $K$ variations and underestimate the irregularities and potential statistical structure of natural heterogeneity, which may cause consistent underestimation of contaminant spreading in such systems (Painter and Mathinthakumar, 1999).

Previous research has shown that spatial variations of $K$ or ln($K$) often display scaling that has been modeled using the properties of fractals (Molz and Boman, 1993; Painter and Paterson, 1994; Painter, 1996a, 2001; Molz et al., 1997). The various mono-fractal models used were the Gaussian fractals, fractional Brownian motion (fBm) and fractional Gaussian noise (fGn), and the non-Gaussian fractal, truncated fractional Levy motion (fLm). In order to employ such fractals, it is desirable to have software for an efficient generating algorithm that is easy to understand and use. Developing and presenting software for such an algorithm is the main objective of the present communication. (Lu et al. (2002) recently used such software in conjunction with their fractal/facies concept to generate complex multi-modal images. The topic of multifractal scaling will not be considered herein (Liu and Molz, 1997; Boufadel et al., 2000), but an interesting perspective may be found in Painter and Mathinthakumar (1999). For a recent detailed review of fractals in subsurface hydrology see Molz et al. (2002).

There are several ways to generate fBm or truncated fLm using such techniques as midpoint displacements methods, successive random additions (SRA) methods (Voss, 1988; Saupe, 1988), Fourier filtering methods (Saupe, 1988), modified turning band methods (Yin, 1996) and the technique known as fractional integration (Mandelbrot, 1983; Tatom, 1995). Although a number of 1- and 2-D algorithms for fBm generation, such as the midpoint displacement algorithm, successive random additions algorithm and Fourier filtering algorithm by Saupe (1988), are available to users, 3-D algorithms for both fBm and truncated fLm generation, which in many applications are needed, are rarely detailed in the literature.

Maeder (1995) outlined an algorithm based on the use of the Mathematica Software Package to simulate 3-D fBm. This might be a practical option for individuals who are familiar with the Mathematica code. Painter (1998, 2001); Painter and Mathinthakumar (1999) developed and applied an algorithm based on generalized sequential simulation to generate truncated fractional Levy motions, fBm and its new variance subordinated flexible scaling model. This algorithm, which is available from the Southwest Research Institute for research applications (Painter, 2002, personal communication), appears to be efficient and powerful. Yin (1996) generalized a modified turning bands method to the 3-D case and provided a brief outline of an implementing algorithm. This method may generate a more accurate 3-D fBm with artifacts stemming from the finite number of lines and the discretization along these lines. Although such artifacts can be minimized effectively by using many lines and points, the computational requirements of this method are demanding.

Research published recently (Caccia et al., 1997; McGaughey and Aitken, 2000) has shown that the common fBm generation methods, such as SRA and the spectral methods (Voss, 1988; Saupe, 1988), do not generate perfect samples of fBm or the associated fGn. The spectral methods are approximate because fBm is non-stationary, and therefore, does not possess a time (or space) independent spectrum. Thus, a spectral density function derived by time or space averaging will be approximate (Yin, 1996; Falconer, 1990). The main problem with the SRA process, is that for an fGn series generated from differencing an SRA-generated fBm series, the auto-correlation for the first few lags will vary from the theoretical correlation, because the fGn at the smaller scale SRA truncation point will not be perfectly stationary (McGaughey and Aitken, 2000). Caccia et al. (1997) suggest a relatively new method called fractional Gaussian process generation, or fGp (Davies and Harte, 1987). To the authors’ knowledge, however, three-dimensional fGp computer algorithms are not yet available. In addition, the uncertainty in most geologic data, such as hydraulic conductivity data, is compatible with a fractal generation algorithm that is not perfect in the sense of reproducing the exact long-range correlation associated with the selected Hurst coefficient. For larger sample lags, the slightly non-stationary behavior of the fGn becomes less obvious. Large-lag correlations are potentially more important than those of small-lags for the generation of a $K$ or ln($K$) field. Also, the SRA algorithm may be truncated arbitrarily well below any lag of practical interest. Therefore, the SRA method, which has an attractive computational speed (an order of $N$ operations to generate $N$ points (Voss, 1988)) and is conceptually simple, is considered in the following development. In addition, the present authors are studying the basics of the SRA process and developing improvements, which we plan to incorporate into future versions of the 3-D algorithm. Furthermore, truncated fLm generation (Painter, 1996a, b) also is incorporated in the algorithm software, with the associated stationary noises, fGn and truncated fLm, being generated by computing the increments of the motions (Molz et al., 1997). An additional benefit of SRA is that being familiar with an intuitive generation algorithm helps one understand the geometry and scaling properties of stochastic fractals. To the authors’ knowledge, software for a simple and efficient 3-D algorithm of generating at least partly anisotropic fBm and truncated fLm is not available in the public domain.
2. Properties of stochastic fractals

In the mono-fractal framework, 1-D fBm is defined as a random, continuous, and single-valued function, \( V(x) \), of the independent spatial or temporal variable \( x \), having stationary increments \( [V(x + h) - V(x)] \) over the distance (lag) \( h \), with a Gaussian distribution of increments for any \( h \). The increments of fBm constitute fGn (Molz et al., 1997). In terms of the mean and variance of the increments, the statistical self-affinity property of fBm, which is different from geometrical self-similarity, may be expressed as
\[
\langle V(x) - V(x + rh) \rangle = 0
\] (1)
and
\[
\langle (V(x) - V(x + rh))^2 \rangle = r^{2H} \langle (V(x) - V(x + h))^2 \rangle, \tag{2}
\]
where \( \langle F \rangle \) represents the expected value of \( F \), and \( r, h \) and \( H \) (\( 0 < H < 1 \)) are constants.

Defining the variances
\[
\sigma_{rh}^2 = \langle (V(x) - V(x + rh))^2 \rangle
\]
and
\[
\sigma_h^2 = \langle (V(x) - V(x + h))^2 \rangle
\]
allows Eq. (2) to be written as
\[
\sigma_{rh}^2 = r^{2H} \sigma_h^2 \Rightarrow \sigma_{rh} = r^H \sigma_h. \tag{3}
\]
The symbols \( \sigma_{rh}^2 \) and \( \sigma_h^2 \) represent the variances of the increments of fBm with lags \( rh \) and \( h \), respectively, and \( \sigma \) represents the corresponding standard deviations. Eqs. (1) and (2) also express what has been termed the scaling properties of fBm. Thus “scaling” has a very specific mathematical meaning within the context of stochastic fractals.

Similarly, a 1-D fLm, which may be viewed as a generalization of fBm (Painter and Paterson, 1994; Painter, 1996a, b; Molz et al., 1997), is defined as a random, continuous, and single-valued function, \( W(x) \), of the independent spatial or temporal variable \( x \), having stationary increments with a Levy-stable distribution. Then fLm is defined as the increments of fLm. A Levy-stable distribution may be defined as the inverse Fourier transform of its characteristic function (Fourier transform), since in general the density distribution does not have a closed form (Samorodnitsky and Taqqu, 1994), i.e.,
\[
f(p) = \frac{1}{\pi} \int_0^\infty \exp(-|Ck|^\alpha) \cos(kp) \, dk, \tag{4}
\]
where \( C \) is a scale parameter and \( \alpha \) is the Levy index. Such a distribution does not have a defined variance when \( \alpha < 2.0 \), nor a defined mean when \( \alpha < 1.0 \). For \( \alpha = 2 \) the Levy distribution becomes a Gaussian distribution. The statistical self-affinity property of fLm may be expressed, in analogy with Eq. (3), as (Samorodnitsky and Taqqu, 1994)
\[
C_{rh}^a = C_h^{a} r^{aH} \Rightarrow C_{rh} = C_h r^{H}, \tag{5}
\]
where \( r \) is the lag distance, \( C_{rh} \) is the scale parameter for the lag distance \( rh \) and \( C_h \) is the scale parameter for the lag distance \( h \). For \( \alpha = 2 \), Eq. (5) becomes identical to Eq. (3), thus showing the scaling analogy between the Gaussian and Levy cases. Standard deviation is related to the “width” of a distribution through the second moment, while the Levy \( C \) is related to “width” through the \( \alpha \) moment, the highest moment that exists for a Levy distribution of index \( \alpha \). This analogy is used in SRA to generate the respective fractal motions.

3. Development of SRA algorithms for fBm and truncated fLm

Before describing the detailed 3-D SRA algorithm, we review the basic idea behind the SRA method (Voss, 1988; Saupe, 1988) and then proceed to the 3-D fractal generation algorithm, including a practical conditioning procedure.

3.1. Review of the SRA method

Based on the fundamental scaling properties of fBm (Eqs. (2) and (3)), Voss (1988) developed a 1D SRA algorithm. Saupe (1988) developed a 2D SRA algorithm and the associated FORTRAN code for square lattices. The basic idea of 1D SRA is illustrated in Fig. 1 (Molz and Boman, 1993). For readers’ convenience, a detailed 1D SRA algorithm is presented here.

One begins with 2 points separated by a distance, \( L \), having initial values of zero. Assume that one wants to generate an fBm with \( NN = 2^n + 1 \) data points. Then the following steps are required:

1. First add random numbers from \( N(0, \sigma_0^2) \) to the 2 end points, denoted by number 1, where \( N(0, \sigma_0^2) \) stands for a Gaussian random number generator with a mean of 0 and a variance \( \sigma_0^2 \).

\[
\begin{array}{c}
\text{Step 1:} \\
1 & L & 1 \\
\end{array}
\]
\[
\begin{array}{c}
\text{Step 2:} \\
1 & 2 & 1 \\
\end{array}
\]
\[
\begin{array}{c}
\text{Step 3:} \\
1 & 3 & 2 & 3 & 1 \\
\end{array}
\]
\[
\begin{array}{c}
\text{Step n:} \\
1, n, n, n, 3 & n, n, n & 2 & n, n, n & 3 & n, n, n & 1 \\
\end{array}
\]

Fig. 1. Schematic diagram illustrating steps used in 1-D SRA method.
2. Linearly interpolate the midpoint (denoted by number 2) value based upon the two end point values resulting from step 1.

3. Then add random numbers from \(N(0, \sigma_1^2)\) with

\[
\sigma_1^2 = (1 - 2^{2H-2}) \frac{\sigma_0^2}{(2^2)^{2H}}
\]

to the midpoint and all other points;

4. Then, linearly interpolate between the 3 points obtained in step (3), denoted by number 3;

5. Add random numbers from \(N(0, \sigma_1^2)\) with

\[
\sigma_2^2 = \frac{\sigma_1^2}{(2^2)^{2H}}(1 - 2^{2H-2})
\]

to the two midpoints interpolated in step (4) and all other points.

6. Repeating the same process, keep performing linear interpolations and adding random numbers from \(N(0, \sigma_1^2)\) with

\[
\sigma_n^2 = \frac{\sigma_{n-1}^2}{(2^2)^{2H}}(1 - 2^{2H-2})
\]

up to the \(n\)th level, which results in a total of \(2^n + 1\) points.

To understand the origin of the variance selection formula that first appears in step 3, consider the initial two points located for convenience at \(x = 0\) and \(L\). Then if we assign random numbers, \(R(0)\) and \(R(L)\), to the two initial points from \(N(0, \sigma_0^2)\), the ensemble of all such points will have a point variance \(\sigma_0^2\) about the mean of 0, and the initial increment variance, \(R(0) - R(L)\), will be \(2\sigma_0^2\), or twice the size of the point variance, which can be easily proved mathematically. Let \(x_1, x_2, x_3, \ldots, x_n\) be independent random numbers from \(N(0, \sigma_0^2)\), then the \(n-1\) increments are \(x_1 - x_2, x_2 - x_3, \ldots, x_{n-1} - x_n\). The variance of the increments becomes

\[
\sigma^2 = \frac{\sum_{i=2}^{n} (x_{i-1} - x_i - \mu)^2}{(n-1) - 1} = \frac{\sum_{i=2}^{n} x_{i-1}^2}{(n-1) - 1} - 2 \sum_{i=2}^{n} \frac{x_i x_{i-1}}{(n-1) - 1} + \sum_{i=2}^{n} \frac{x_i^2}{(n-1) - 1} = \sigma_0^2 + \sigma_0^2 = 2\sigma_0^2,
\]

where \(\mu\) is the mean of the increments and equals 0. Since the random numbers \(x_i, x_{i-1}\) are independent, the term

\[
\sum_{i=2}^{n} \frac{x_i x_{i-1}}{(n-1) - 1} = \sum_{i=2}^{n} \frac{x_i}{(n-1) - 1} \cdot \sum_{i=2}^{n} \frac{x_{i-1}}{(n-1) - 1} = 0.
\]

(It is important to distinguish between the point and increment variances, because the fBm scaling rule applies to the increment variance.) If we now linearly interpolate the ensemble of points, all increments will be reduced by a factor of 2, so the resulting increment variance will be reduced by a factor of 4 to become \(2\sigma_0^2/4 = \sigma_0^2/2\), and the new point variance will be half of this, or \(\sigma_0^2/4\). If a random number from \(N(0, \sigma_1^2)\) is now added to all points, the new point variance will be that resulting from the interpolation plus what is added, or \(\sigma_0^2/4 + \sigma_1^2\). This will yield an increment variance of \(\sigma_0^2/2 + 2\sigma_1^2\), and one must select \(\sigma_1^2\) so that the increment variance scaling relationship represented by Eq. (3) is satisfied. This requires \(\sigma_0^2/2 + 2\sigma_1^2 = 2\sigma_0^2/2^{2H}\), or \(\sigma_1^2 = \sigma_0^2(1 - 2^{2H-2})/2^{2H}\), which is the formula contained in step 3. From now on, one may use the scaling relationship for the increment variance given by \(2\sigma_{n+1}^2 = 2\sigma_n^2/2^{2H}\), which yields the general variance scaling formula given in step 6 as \(\sigma_n^2 = \sigma_0^2(1 - 2^{2H-2})/2^{nH}\). Note that \(\sigma_n^2\) alone always represents the variance of the PDF used to generate the random numbers to be added to the point values at step \(n\). Thus, it represents an added point variance. The final result is a realization of the random function, fBm, with a Hurst coefficient \(H\).

As illustrated in Fig. 2, one may initiate 2D SRA by starting with a single square of four corner points which are denoted by number 1, having random numbers assigned from \(N(0, \sigma_0^2)\). Therefore, the variance of increments between these points is \(2\sigma_0^2\). Then, we linearly interpolate the center point, denoted by number 2, of the square with the four corner values. Thus, the increment variance and point variance of this new point should be \(2\sigma_0^2/4\) and \(2\sigma_0^2/8\), respectively. Now, adding random numbers from \(N(0, \sigma_1^2)\) to all points, we get a point variance of \(2\sigma_0^2/8 + \sigma_1^2\) for the new center point. Applying the scaling rule for the increments, we have \(2(2\sigma_0^2/8 + \sigma_1^2) = (1/\sqrt{2})2^{2H}2\sigma_0^2\), that is, \(\sigma_1^2 = (1/2)^H\sigma_0^2(1 - 2^{2H-2})\). The next step is to linearly interpolate the midpoints,
denoted by number 3, of each side of the square by using
two adjacent corner values and then add random
numbers from \( N(0, \sigma^2_{e}) \) with \( \sigma^2_{e} = (1/2)^H \sigma^2_{e} \) to all points.
Following the same logic, after \( n \) steps a 2-D \( \mathbf{fBm} \) with
the Hurst coefficient \( H \) and a total of \( (2^n + 1)(2^n + 1) \)
data points is obtained. It should be pointed out that we
added a factor \( (1 - 2^{H - 2}) \) to the variance of the
displacements in Saupe’s 2-D pseduo-code (1988). This
corrects an inconsistency in the first step of the algorithm.

3.2. Generation of 3D \( \mathbf{fBm} \)

The basic scaling property represented by Eq. (3) is
again used to develop an SRA algorithm for generating
\( \mathbf{fBm} \) in 3D. To develop the algorithm, a cube with a unit
diameter \( (\sqrt{2}/2)^3 \) is used as a starting point. The
general idea is to first divide the cube with the edge
length \( 1 \) into eight sub-cubes with edge lengths of \( 1/2 \).
Proper interpolations and random additions are applied
to each point of sub-cube. Then each sub-cube is divided
into another set of eight sub-cubes with the edge length
of \( 1/2^2 \). This dividing process is continued down to the
desired scale. By the following detailed procedure given
below, a 3D \( \mathbf{fBm} \) with a desired \( H \) value is obtained.

1. Assign random numbers from \( N(0, \sigma^2_{0}) \) to the cube
corners (node types 1 in Fig. 3).
2. Interpolate the central point value of the cube by
arithmetic averaging of the eight corner values (node
types 2 in Fig. 3).
3. Add random numbers from \( N(0, \sigma^2_{3,1}) \) with
\[
\sigma^2_{3,1} = \sigma^2_{0} \left( \frac{\sqrt{3}}{2} \right)^{2H} \left( 1 - \frac{1}{4} \left( \frac{3}{2} \right)^H \right)
\]
(note: this scaling relationship and those given in
steps 3, 5 and 7 can be obtained if one follows the
logic given in Section 3.1) to all points.
4. Interpolate the midpoints of the cube edge (nodes
type 3) by arithmetic averaging of the corner values
that are located on the same edge;
5. Add random numbers from \( N(0, \sigma^2_{3,1}) \) with
\[
\sigma^2_{3,1} = \sigma^2_{0} \left( \frac{1}{\sqrt{2}} \right)^{2H} \left( 1 - \frac{1}{4} \left( \frac{3}{2} \right)^H \right)
\]
to the midpoints and all other points.
6. Interpolate the central points of the cube face by
arithmetic averaging of the four corner values located
on the same face (node types 4 in Fig. 3);
7. Add random numbers from \( N(0, \sigma^2_{3,1}) \) with
\[
\sigma^2_{3,1} = \sigma^2_{0} \left( \frac{1}{\sqrt{2}} \right)^{2H} \left( 1 - \frac{1}{4} \left( \frac{3}{2} \right)^H \right)
\]
to the central points and all other points;
8. The original cube is now divided into eight sub-
cubes. For each sub-cube, apply steps 2 through step
7. However, the scaling rules in steps 3, 5 and 7 will be
\( \sigma^2_{3,1} = \left( \frac{1}{2} \right)^{2H} \sigma^2_{3,1} \), \( \sigma^2_{3,1} = \left( \frac{1}{2} \right)^{2H} \sigma^2_{3,1} \), and \( \sigma^2_{3,1} = \left( \frac{1}{2} \right)^{2H} \sigma^2_{3,1} \) (where \( n = 2 \), respectively.
9. Following the same process, repeat step (8) on the
each sub-cube up to level \( n \), which will result in a 3-D
\( \mathbf{fBm} \) with a Hurst coefficient \( H \) and a total number of
points \( (2^n + 1) \times (2^n + 1) \times (2^n + 1) \).

Since \( \mathbf{fBm} \) and \( \mathbf{fLm} \) are not functions that have
natural geometric bounds, minor inconsistencies in
scaling and increment stationarity will exist mostly at
generation domain boundaries and at the starting and
stopping points of the algorithm. If desired, these

---

![Schematic diagram illustrating steps used in 3-D SRA method.](image-url)
inconsistencies can be diminished by generating a realization in a domain that is larger than the desired domain, starting the algorithm at an $L$ scale above the application scale desired and stopping it at a scale well below that desired. Then an appropriate interior portion of the generated function with desired starting and stopping scales is selected as the actual realization.

3.3. Anisotropy with fractal $K$ processes

Anisotropic porous media are widely observed. Often little is known about the statistical properties of vertical hydraulic conductivity, so an anisotropic porous medium is commonly modeled by using different magnitudes of hydraulic conductivity in two or three principal directions, with the main difference being in the vertical versus horizontal directions. However, stochastic fractal representations of $K$ offer the possibility of more complex types of anisotropy. For example, one would expect the possibility of different variance scaling and autocorrelation in the vertical versus horizontal directions, as reflected by different $H$ values and $\sigma_0$ values, even for the horizontal hydraulic conductivity (Molz et al., 1997). The 3D SRA algorithm described herein provides a way to generate geometrical anisotropic porous media with the same scaling parameter $H$ in both directions but different variances. For example, to obtain a geometric anisotropic 3-D fBm realization, with the variance of $\ln(K)$ increments in the vertical direction four times that of the increments in the horizontal direction and a Hurst coefficient $H$ of 0.3, one can first generate an isotropic 3-D fBm realization with total, say, $256 \times 256 \times 256$ ($x \times y \times z$) grid points; then discard all but 1 layer out of every 10 layers of the isotropic 3-D fBm realization. The resulting realization has total $256 \times 256 \times 25$ grid points with the variance in the direction four times that in the horizontal direction. This is because the new variance for the lag distance 1 in the vertical direction equals $\sigma_1^2 = \sigma_l^2 H \approx 4\sigma_1^2$ if $l = 10$ and the Hurst coefficient $H = 0.3$. This approach is similar to the rescaled (stretch) distance method suggested by Painter (1996a, b).

In applications, node systems with uneven spacing are often required. For example, to simulate flow fields accurately and efficiently in a pump and treat system, one usually needs finer grid sizes around the well and larger grid sizes elsewhere. To obtain such a grid system, one may start with a fractal realization with the finer grid size resolution and then use various averaging schemes (e.g. harmonic averaging) to obtain $K$ values for bigger sized blocks where it is needed, based on the detailed finer size grid system. Because the proposed algorithm is so efficient, such a task can be achieved easily.

3.4. Generation of 3D, truncated, fractional levy motion

For fLm generation, the scaling property of fLm, Eq. (5), may be used as a basis for selecting appropriate random numbers. As discussed in Painter (1996a,b), truncated Levy-stable distributions must be used in order to obtain realistic $K$ realizations. Although, mathematically speaking, the truncation may degrade the scaling properties of the random field; numerical experiments (see Section 4) show that the resulting simulations have approximate Levy fractal scaling properties. The generation procedure for 1-D fLm is identical to that for 1-D fBm, except that $\sigma$ in the algorithm given previously is replaced by $C$, the exponent 2 is replaced by $x$ (i.e., $\sigma^2$ is replaced by $C^x$), and a truncated Levy-stable random number generator is used in place of a Gaussian generator (Painter, 1996a,b). In order to achieve effective truncation, one rejects any random number greater than a preset bound. Due to the truncation, the variance of the increments of a truncated fLm realization is finite. The analogy carries over to the multi-dimensional generation procedures.

Similar to the method for obtaining a geometric anisotropic 3-D fBm realization described in the Section 3.3, a geometric anisotropic 3-D fLm realization can be easily generated. For example, to obtain a truncated fLm realization with, say, a Hurst coefficient $H = 0.3$ and $C_2 = 2C_b$ ($C_e$ and $C_b$ are the scale parameter in the vertical and horizontal directions, respectively), one can first generate an isotropic 3-D truncated fLm realization, then discard all but 1 layer out of every 10 layers of the isotropic fLm realization. This is simply because the new scale parameter for the lag distance 1 in the vertical direction equals $C_2 \approx C_1^{0.5} \approx 2C_b$ if $l = 10$ and the Hurst coefficient $H = 0.3$.

3.5. A practical conditioning procedure

In subsurface hydrological applications, $K$ measurements are required in order to determine the Hurst coefficient $H$ and other statistical properties. Since the measurements are a direct tie to reality, one should take advantage of the measured values when constructing a realization. A conditioned realization, which is one that honors measured data at the sample locations while preserving the selected fractal structure, is highly desired.

There are several possible ways to construct conditioned realizations. Painter (1998) used the sequential simulation algorithms utilizing the numerically constructed conditional density to reproduce the desired statistical properties in simulations. Hewett and Behrens (1988) used the method of residuals to obtain conditioned realizations that involved Kriging (Journel and Huijbregts, 1978) as an interpolation method. Herein, we use the simple and efficient inverse-distance-weighting...
procedure (IDWM) (e.g. Molz et al, 1998), rather than Kriging, to interpolate data approximately in applications.

To initiate the calculation of residuals (Journel and Huijbregts, 1978), a smooth curve, called W₁ (Fig. 4a), is fitted to the measured data using IDWM. Then an unconditioned \( \ln(K) \) realization, called W₂ (Fig. 4b), with the selected \( H \) value and variance extracted from the measured data set, is generated using the 3-D SRA algorithm described previously. Next, another smooth curve, called W₃ (Fig. 4c), is fitted to the realized points that fall at the measurement locations using IDWM. Finally, the conditioned \( \ln(K) \) realization, \( W \), is obtained by computing \( W = W₁ + W₂ - W₃ \). The resulting curve \( W \) reproduces the data values at locations where measurements were made and embodies a good approximation of the statistical properties of the measured data set, including the \( H \) value and variance. This process will be illustrated in the next section.

4. Numerical simulations and validations

Based on the fractal generation procedures described earlier, a Fortran code, which includes 3-D fBm, truncated fLm and a 3-D fractal conditioning algorithm, is developed. To generate an unconditioned 3-D geometric anisotropic fBm realization, we input a Hurst coefficient \( H = 0.3 \) that is near the center of reported \( H \) values for \( \ln(K) \) (Neuman, 1990), a variance \( \sigma_H^2 = 1 \) and a ratio of the vertical \( \ln(K) \) increment variance to the horizontal \( \ln(K) \) increment variance, selected as 4. A portion of the resulting realization, which clearly has anisotropic features, is shown in Fig. 5a. Dispersional analysis (Bassingthwaighte, 1988; and for more information, see Appendix A), which is applicable only to fGn, is used to analyze the increments of fBm for the resulting \( H \) value. An analysis based on an average of 100 1-D horizontal \( \ln(K) \) increment samples of length 256 extracted from the above realization shows that the Hurst coefficient \( H = 0.41 \) (Fig. 5b), which is larger than the input value of 0.3. To obtain the desired \( H = 0.3 \).
value, one may start with a smaller $H$ value; with a little trial-and-error, the desired $H$ value may be obtained after a few runs. The variance of increments for lag = 1 is equal to 1.0, which is achieved by rescaling the increments of the realization to the increments with the designed variance 1.0 in the code.

Several studies (Bassingthwaighte and Raymond, 1995; Cannon et al., 1997; Caccia et al., 1997; Eke et al., 2000) indicate that dispersive analysis (Bassingthwaighte, 1988) is the most reliable and accurate test for detecting variance-scaling. A brief introduction to dispersive analysis and a comparison with other techniques may be found in Appendix A. Dispersional analysis, therefore, is used to analyze fBms generated with the proposed 3-D SRA algorithm for $H$ values presented in Table 1. The results show that the 3-D SRA algorithm produces $H$ values larger than those selected when $H < 0.5$, and smaller values than those selected when $H > 0.6$. Such discrepancies can be easily corrected with several trial runs.

For an example of an unconditioned 3-D geometric anisotropic truncated fLm realization, we use a Hurst coefficient $H = 0.35$, a scale parameter $C = 0.1$, a Levy index $\alpha = 1.7$ and a ratio of the vertical scale parameter to the horizontal scale parameter of 2. A portion of the realization is shown in Fig. 6a). Although strictly speaking, the truncation ($\pm 5$ is applied in this case) may degrade the fractal scaling feature, numerical tests show that the resulting simulation has an approximate power correlation structure (Eq. (5)) with a slightly larger $H = 0.41$ as shown in Fig. 6b). The Levy index, $\alpha$, calculated using the method of Fama and Roll (1971) equals 1.75 for lag = 1, which is slightly larger than the input value of 1.7, and as shown in Fig. 6c), the increment probability density plot for a lag of 1 is fitted well within the truncation limits of $\pm 5$ by a Levy probability density function having a scale parameter of $C = 0.1$ and a Levy index $\alpha = 1.75$. (The estimates of $H$ and $\alpha$ are based on a data set of twenty 1-D horizontal $\ln(K)$ samples extracted from the above realization, each containing 257 $\ln(K)$ values.)

For additional illustration, a 3-D fBm realization, which is conditioned on measured $\ln(K)$ data from the MADE site (Boggs et al., 1992), is presented in Fig. 7a). The input $H$ and $\sigma^2$, which are extracted from the MADE data set, are 0.3 and 1.0, respectively. In this realization, a ratio of the vertical $\ln(K)$ increment variance to the horizontal $\ln(K)$ increment variance

<table>
<thead>
<tr>
<th>Table 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dispersional analyses (DA) of fBms generated with SRA algorithm</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Input $H$</td>
</tr>
<tr>
<td>0.1</td>
</tr>
<tr>
<td>0.2</td>
</tr>
<tr>
<td>0.3</td>
</tr>
<tr>
<td>0.4</td>
</tr>
<tr>
<td>0.5</td>
</tr>
<tr>
<td>0.6</td>
</tr>
<tr>
<td>0.7</td>
</tr>
<tr>
<td>0.8</td>
</tr>
<tr>
<td>0.9</td>
</tr>
</tbody>
</table>

$^a$ $H$ values obtained by dispersive analysis are based on 100 1-D horizontal $\ln(K)$ increment samples of length 256 extracted from each realization.
was selected as 4. The selected $H$ value is approximately reproduced, that is, $H = 0.29$, after two trial runs; and the variances in both the vertical and horizontal directions are exactly produced with a ratio of 4. Then the conditioning procedure described previously is applied to obtain the final realization. The nearest 10 points have been used to interpolate the values at un-sampled locations in the IDWM. Numerical tests show that the scaling property is approximately obtained with $H = 0.30$ (Fig. 7b)) and the variance is well preserved.

5. Conclusions

A 3-D approach is often necessary to simulate plume migration in heterogeneous media. Therefore, software for an efficient algorithm that can produce $\ln(K)$ realizations with selected statistical properties is developed. In practice, fBm or truncated fLm models can be used to approximately simulate logarithmic hydraulic conductivity $\ln(K)$ spatial variation. The 3-D SRA algorithm described herein, which is easy to understand and has a very attractive computational speed, with an order of $N$ operations for $N$ nodes, can be used to generate geometric anisotropic fBm/fLm in three dimensions. The resulting fractal scaling properties have been validated numerically for both fBm and truncated fLm.

In fBm simulation, not only is the scaling property approximately observed, but the exact variance is obtained through a rescaling procedure in the code. In fLm simulation, due to the truncation utilized in the fLm algorithm and the central limit theorem, the Levy-stable index derived from the realizations is slightly larger than the input $\alpha$ value. However, the sample probability density is well approximated by the theoretical Levy probability density distribution up to the truncation of $\pm T$. A slight disadvantage of the method is that several trial runs are usually needed to obtain a pre-selected value for the scale parameter $C$.

It should be pointed out that the SRA method suffers a minor problem, that is, the correlation of the increments of a realization for the first two lags are known to vary from the desired correlation, because the increments for these small lags are not perfectly stationary. However, for larger sample lags, the slightly non-stationary behavior of the fGn becomes less obvious (McGaughey and Aithen, 2000). Once again, a pre-selected $H$ value may be obtained using a few trial runs.
Conditioning in constructing fractal realizations is an important step in the sense that this step forces the final fractal realization to go through the valuable measured data. Numerical tests show that the proposed conditioning procedure can approximately preserve the scaling property of the observed data. Computationally, this conditioning procedure is also fast; therefore, it is useful for multiple large hydraulic conductivity simulations in applications.

Recently, new one-dimensional versions of fBm construction and testing procedures have been introduced (Davies and Harte, 1987; Bassingthwaighte and Raymond, 1995; Caccia et al., 1997; Cannon et al., 1997; Eke et al., 2000). Both focus on direct generation and testing of fGn, which makes sense since that represents the stationary process. The generation procedure is called fractional Gaussian process generation (Davies and Harte, 1987; Caccia et al., 1997), and the testing procedure is called dispersional analysis (Bassingthwaighte and Raymond, 1995; Caccia et al., 1997). Dispersional analysis was employed in the present work to calculate the \( H \) values resulting from the various fBm realizations, and is included as one of the computer codes associated with this communication. A useful objective for future research would be to develop higher dimensional versions of fractional Gaussian process generation, since the existing 1-D version is superior to the corresponding version of SRA (Caccia et al., 1997).

### Acknowledgements

This research has been supported by Grant No. R82-6171 from the US Environmental Protection Agency’s Science to Achieve Results (STAR). However, it has not been subjected to EPA review and therefore does not necessarily reflect the views of the agency, and no official endorsement should be inferred.

### Appendix A. Dispersional analysis

Several techniques for detecting fractal scaling, such as rescaled range analysis, variogram analysis, and power spectral analysis, have been used in hydrogeology. Among them, rescaled range analysis was thought to be one of the more reliable methods (Mandelbrot and Wallis, 1966). However, several recent studies have shown that in general, rescaled range analysis is not dependable, particularly for smaller sample sizes (<10^5) and Hurst coefficients >0.5 (Cannon et al., 1997; Caccia et al., 1997). In many applications, there are only a few hundred to several thousand field or lab sample data available. Results using rescaled analysis on such data sets are often misleading (Caccia et al., 1997).

Dispersional analysis (Bassingthwaighte, 1988) is based on the variability of local averages of fGn over windows of length \( L = n \Delta x \), where \( \Delta x \) is the smallest spatial resolution (lag) of the data. It uses the standard deviation, SD, of these local averages, and repeats the calculation for many different \( L \) values. \( H \) equals 1 plus the slope of the regression of \( \log(\text{SD}(L)/\text{SD}(L_0)) \) versus \( \log(L/L_0) \), where \( L_0 \) is a reference window size. Bassingthwaighte (1988) shows that

\[
\log \text{SD}(L) = (H - 1) \log(L/L_0) + \log \text{SD}(L_0)
\]

or

\[
\text{SD}(L) = \text{SD}(L_0)(L/L_0)^H - 1.
\]

For the purpose of comparison between the various techniques for estimating \( H \), nine exact fractional Gaussian process (fGp) series, each having 4096 data points and a different \( H \) value, were generated using the methods of Davies and Harte (1987). Then the several \( H \)-estimating procedures are applied to each individual fGp series and the results are presented in Table 2. Our results confirm that dispersional analysis, on the average, is superior to the other methods tested, which is similar to the conclusion of the more detailed study presented by Caccia et al. (1997).

<table>
<thead>
<tr>
<th>Methods</th>
<th>Rescaled range analysis</th>
<th>Variogram analysis</th>
<th>Power spectral analysis</th>
<th>Dispersional analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>fGp ( H ) values</td>
<td>0.16</td>
<td>0.059</td>
<td>0.089</td>
<td>0.13</td>
</tr>
<tr>
<td></td>
<td>0.27</td>
<td>0.21</td>
<td>0.095</td>
<td>0.21</td>
</tr>
<tr>
<td></td>
<td>0.40</td>
<td>0.40</td>
<td>0.23</td>
<td>0.31</td>
</tr>
<tr>
<td></td>
<td>0.42</td>
<td>0.45</td>
<td>0.46</td>
<td>0.39</td>
</tr>
<tr>
<td></td>
<td>0.43</td>
<td>0.37</td>
<td>0.53</td>
<td>0.46</td>
</tr>
<tr>
<td></td>
<td>0.56</td>
<td>0.45</td>
<td>0.57</td>
<td>0.55</td>
</tr>
<tr>
<td></td>
<td>0.61</td>
<td>0.77</td>
<td>0.73</td>
<td>0.73</td>
</tr>
<tr>
<td></td>
<td>0.64</td>
<td>0.76</td>
<td>0.89</td>
<td>0.80</td>
</tr>
<tr>
<td></td>
<td>0.71</td>
<td>0.72</td>
<td>0.92</td>
<td>0.82</td>
</tr>
</tbody>
</table>
References


