3. RANDOM FIELD GENERATORS

3.1 Introduction

The generation of spatially correlated samples of random fields plays a fundamental role in the numerical analysis of stochastic functions - whether these are 1-, 2-, or 3-dimensional. The purpose of random field simulation is to create numerical samples or "realizations" of stochastic processes with well-defined properties. The term "random field generator" is actually improper, because random fields are by definition probability spaces (see chapter 2) and can therefore not be discretely generated. For ease of reading and in reference to many other publications that deal with the generation of random realizations of a random field, the term "random field" is in this and all subsequent chapter used interchangeably with the term "realization". The random fields as defined in section 2.5.1 are henceforth referred to as random field variables (RFVs).

The simplest and most commonly available form of simulation is the random number generator on a calculator or computer. These readily accessible simulators generate independent, uniformly distributed random numbers i.e., samples of a single random variable X with a uniform, univariate distribution (e.g. Press et al., 1992). If X is not uniformly distributed it is a relatively easy task to transform these random numbers such that they follow any other desired univariate distribution.

The simplest case of a random field variable (random process) is an orthogonal RFV, which consists of random univariate samples at each location. This can be implemented easily with any good random number generator. A particular challenge arises, however, when the random variables $X_i=X(\mathbf{x}_i)$, $X_j=X(\mathbf{x}_j)$ $(i\neq j)$ are dependent i.e., when they are (spatially) correlated and defined through a joint or multivariate distribution. Not only do the generated random fields have to converge in the mean square to the desired ensemble mean and variance (and any higher order moments if appropriate), they also have to converge in the mean square to the

desired correlation structure as the number of samples increases. In this chapter, algorithms are introduced that generate such random fields.

In practice the joint probability distribution function is often inferred from field data obtained at the site of interest. The joint probability distribution is commonly described by invoking the ergodicity and stationarity hypotheses discussed in the previous chapter and by taking the sample mean and sample variance-covariance functions as the moments of the underlying multivariate pdf. To take full advantage of the field data the simulations must be conditioned on the information known about the particular points in space, where measurements were taken. This amounts to the generation of random variables with a conditional joint probability distribution function. The ensemble of conditional realizations is a subset of the ensemble of unconditional realizations. The conditional subset consists of all those samples in the unconditional set, that preserve the known data at the measured locations. As shown in the previous chapter the conditional joint distribution of the random variables is different from the unconditional multivariate pdf. The generation of conditional random fields therefore needs to go beyond the capabilities of an unconditional random field generator.

In this chapter several popular random field generators (RFGs) are described and compared. Random number generators (RNGs) are also tested. First an unconditional two-dimensional random field generator based on spectral representation and a fast Fourier transform is introduced. A conditioning method based on kriging estimation is presented next. The statistical performance of the spectral random field generator (SRFFT) is compared with the turning band method (TB), the matrix decomposition method (LU) and the sequential Gaussian simulation method (S). The numerical efficiency of these RFGs has been assessed elsewhere (Tompson et al., 1989).

3.2 (Unconditional) Two-Dimensional Random Field Generation by the Spectral Method

The purpose of a random field generator is to transform an orthogonal realization consisting of independently generated random numbers with a prescribed univariate distribution into a correlated random field with the desired joint probability distribution. If the distribution is Gaussian, the joint pdf is expressed by its first two moments, the mean and the covariance. In the previous chapter a transformation was introduced that is ideally suited for building a random field generator: the spectral representation dZ of a correlated RFV X is itself an RFV of independent random variables with a variance defined by the spectral density function of X, $S(\mathbf{k})\, d\mathbf{k}$. Recall that the spectral density $S(\mathbf{k})$ of X is the Fourier transform of the covariance function $C(\xi)$ of X where ξ is the separation distance. Hence, if random, zero-mean $dZ(\mathbf{k})$ are generated with a variance $S(\mathbf{k})\, d\mathbf{k}$ then their inverse Fourier transform yields a correlated random field with $X(\mathbf{x})$ that have zero-mean and the desired covariance function by virtue of (2-51). Random field generators based on Fourier transforms have first been introduced by Shinozuka (1972, 1991). Gutjahr (1989) describes a two-dimensional random field generator based on a fast Fourier transform algorithm, which has been adopted for our study.

In the previous chapter the spectral representation of a continuous, infinitely large random field was defined. In the numerical generation of random fields, however, one is limited both in the extent of the random field and in the number of points generated. Hence, (2-51) must be restated to accommodate *finite* random fields defined on a countable number of *discrete* grid-points. The following derivations are specifically for two-dimensional random fields. But the extension to higher dimensional random fields should be obvious and is straight forward.

For the purpose of this study realizations are generated on a rectangular domain defined over a regular grid centered around the origin with gridpoints being $\Delta\mathbf{x} = (\Delta x_1, \Delta x_2)^T$ apart. The size of the domain is defined by $M\,\Delta\mathbf{x}$ such that the rectangle spans the area between $-M\Delta\mathbf{x}$ and

$(M-1)\Delta\mathbf{x}$ and the number of gridpoints in the random field is 2M by 2M. Since the spectral representation of a stationary random field is only defined for an infinite domain, it is further assumed that the random field is periodic with period 2M in both dimensions. This has no direct impact on the generated random field. But it is a necessary assumption for the formal derivation of its spectral representation, because the analysis of an infinite process can be used for the generation of a finite random field. There is another reason for choosing the assumption of periodicity (after all, any other values for the random field outside $[-M\Delta\mathbf{x}, (M-1)\Delta\mathbf{x}]$) could have been assumed): Periodic functions are known to have a discrete rather than a continuous spectrum i.e., only a discrete set of frequencies contributes to the spectral representation of the periodic stationary random field. Hence, $dZ(\mathbf{k})$ exists only for discrete $\mathbf{k}$, for which it can be generated such that $\langle dZ(\mathbf{k})\rangle = 0$ and $\langle |dZ(\mathbf{k})|^2\rangle = S(\mathbf{k})\, d\mathbf{k}$.

The discretization of $X(\mathbf{x})$ limits the wavelengths "seen" by the discrete random field to all those that are at least of length $2\Delta\mathbf{x}$ i.e., to all (angular) frequencies $\mathbf{k} \le 2\pi/(2\Delta\mathbf{x})$. Higher frequencies cannot be distinguished from frequencies within this limit, an effect referred to as "aliasing". In other words, heterogeneities on a scale smaller than the discretization $\Delta\mathbf{x}$ are not resolved by the random field. Similarly, the longest possible wavelength "seen" by a finite random field is less than or equal to $2M\Delta\mathbf{x}$ i.e., the lowest (angular) frequency is $\Delta\mathbf{k} = 2\pi/(2M\Delta\mathbf{x})$, and all other frequencies $\mathbf{k}$ must be multiples of $\Delta\mathbf{k}$. Hence, the spectral representation $dZ(\mathbf{k})$ of a finite, discrete random field $X(\mathbf{x})$ with $(2M)^2$ gridpoints in 2-D space is also a finite, discrete random field defined on a $(2M)^2$ grid in the 2-D frequency domain. Note that the discretization in $X(\mathbf{x})$ determines the size of the field of $dZ(\mathbf{k})$, while the finite size of $X(\mathbf{x})$ determines the discretization of $dZ(\mathbf{k})$. For discrete $dZ(\mathbf{k})$ the Fourier-Stieltjes integral (2-49) becomes a Fourier series such that

$$X(\mathbf{x}_n) = \sum_{m=-M}^{M-1} \sum e^{i\mathbf{k}_m \mathbf{x}_n} z(\mathbf{k}_m) \qquad \textbf{(3-1)}$$

where $z(\mathbf{k})$ are (complex valued) random Fourier coefficients with the same properties as $dZ(\mathbf{k})$ in (2-49), namely zero-mean, a variance $\sigma^2_{z(\mathbf{k})} = S(\mathbf{k})\, \Delta\mathbf{k}$, and all $z(\mathbf{k})$ independent for $\mathbf{k}_1 \neq \mathbf{k}_2$.

To ensure that X(**x**) is a real valued random field, the z(**k**) field must be constructed such that

$$z(-\mathbf{k}) = z^*(\mathbf{k}) \tag{3-2}$$

i.e., random numbers z(**k**) need only be generated for one half the size of the rectangle. The * stands for complex conjugate. Complex valued, Gaussian distributed z(**k**) for discrete $\mathbf{k}_j$, $j=1,(2M)^2/2$ are obtained by generating two independent Gaussian random numbers α_j and $ß_j$ for each $\mathbf{k}_j$, each with zero-mean and variance ½, and construct

$$z(\mathbf{k}_j) = (S(\mathbf{k}_j)\ \Delta\mathbf{k})^{1/2}\ \frac{(\alpha_j + i ß_j)}{2} \tag{3-3}$$

for one half of the random field. The other half of the random field is obtained through the symmetry relation (3-2). It can be shown by inspection that the above construction of z(**k**) satisfies the required properties (Gutjahr et al., 1989). After constructing a field z(**k**) by the above method, which merely requires the generation of independent Gaussian distributed random numbers, the correlated random field X(**x**) is obtained by performing the Fourier summation (3-1).

The double summation in (3-1) is most efficiently done by a numerical Fourier transform technique called the "Fast Fourier transform" or simply FFT (Brigham, 1988). FFT algorithms can be found in many computer libraries (e.g. IBM, 1993) and are described in books on numerical mathematics (e.g. Press et al., 1992). It suffices to say that FFT algorithms essentially perform a transformation as (3-1), but in a computationally very efficient manner. Most available FFT algorithms are written using the frequency **u** as argument instead of the angular frequency **k**, where $\mathbf{k} = 2\pi\mathbf{u}$. Recall the following definitions of Fourier transform pairs from chapter 2:

$$C(\mathbf{x}_i) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{i\mathbf{k}\xi}\ S(\mathbf{k})\,d\mathbf{k} \tag{3-4}$$

$$S(\mathbf{k}) = \frac{1}{(2\pi)^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{i\mathbf{k}\xi} \, C(\xi) \, d\xi \qquad \textbf{(3-5)}$$

$$X(\mathbf{x}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{i\mathbf{k}\xi} dZ(\mathbf{k}) \qquad \textbf{(3-6)}$$

$$dZ(\mathbf{k}) = \frac{1}{(2\pi)^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{i\mathbf{k}\xi} \, X(\mathbf{x}) \, d\mathbf{x} \qquad \textbf{(3-7)}$$

Changing the variables of integration from **k** to **u**, where d**k** = 2π d**u**, the above transform pairs become:

$$C(\xi) = (2\pi)^2 \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{i2\pi\mathbf{u}\xi} \, S(2\pi\mathbf{u}) \, d\mathbf{u} \qquad \textbf{(3-8)}$$

$$S(2\pi\mathbf{u}) = \frac{1}{(2\pi)^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{-i2\pi\mathbf{u}\xi} \, C(\xi) \, d\xi \qquad \textbf{(3-9)}$$

$$X(\mathbf{x}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{i2\pi\mathbf{u}x} \, dZ(2\pi\mathbf{u}) \qquad \textbf{(3-10)}$$

$$dZ(2\pi\mathbf{u}) = \frac{1}{(2\pi)^2} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{i2\pi\mathbf{ux}}\, X(\mathbf{x})\, d\mathbf{x} \qquad \textbf{(3-11)}$$

Typical FFT algorithms also require that the summation in (3-1) is over the interval [0,2M-1] rather than over the interval [-M,M-1]. Using the periodicity assumption $z(m\,\Delta\mathbf{k})$, $m > M-1$, are obtained from:

$$z(m\Delta\mathbf{k}) = z((m-2M)\Delta\mathbf{k}), \qquad \text{all } m > M-1 \qquad \textbf{(3-12)}$$

Recalling that $\Delta\mathbf{k} = (2\pi)/(2M\Delta\mathbf{x})$ this leads to the following construction of the correlated random field $X(\mathbf{x})$ with entries $X(n_1\,\Delta\mathbf{x}_1, n_2\,\Delta\mathbf{x}_2)$, $0 \le n_1, n_2 \le 2M-1$:

$$X(n_1\Delta x_1, n_2\Delta x_2) = \sum_{m_1=0}^{2M-1}\sum_{m_2=0}^{2M-1} \exp\left(i\frac{2\pi m_1 n_1}{2M}\right) \exp\left(i\frac{2\pi m_2 n_2}{2M}\right) z(m_1,m_2) \qquad \textbf{(3-13)a}$$

where

$$z(m_1,m_2) = \left[\frac{2\pi}{2M\Delta x_1}\,\frac{2\pi}{2M\Delta x_2}\, S\left(\frac{2\pi m_1}{2M\Delta x_1}, \frac{2\pi m_2}{2M\Delta x_2}\right)\right]^{1/2} (\alpha_{m_1,m_2} + i\beta_{m_1,m_2}) \qquad \textbf{(3-13)b}$$

with α and β being zero-mean, independent, Gaussian distributed random numbers of variance ½. For this study, random fields are generated using (3-13) with the SCFT2 subroutine in the ESSL Fortran library to perform the FFT (IBM, 1993), and with the GAUSDEV and RAN2 subroutines from Press et al. (1992) to generate the random numbers α and β. The original implementation of this random field generator was generously provided by Allan Gutjahr (1989).

3.3 Conditional Two-Dimensional Random Fields

Assume an array of measurements $\mathbf{X}_1 = \{x_1,...,x_m\}$ is available and a two-dimensional conditional random field must be generated such that at locations $\{\mathbf{x}_1,...,\mathbf{x}_m\}$ the measured value

of the random variables $\mathbf{X_1}$ are reproduced with probability 1, and such that at all other locations $\{\mathbf{x}_{m+1},...,\mathbf{x}_n\}$ the generated random numbers $\mathbf{X_2} = \{x_{m+1},...,x_n\}$ have a sample mean and sample covariance that converge in the mean to the conditional mean $<\mathbf{X_2}>^c$ and conditional covariance $\mathbf{E_{22}}$ (see section 2.5.3), respectively, in the limit as the number of random fields generated becomes infinite.

To implement the conditional random field generation, Delhomme (1979) used the following approach based on work by Matheron (1973) and Journel (1974, 1978): Initially, the measured data $\mathbf{X_1}$ are used to infer the moments (mean and covariance) of the unconditional joint pdf of the random field. Then an estimate of the conditional mean $<\mathbf{X_2}>^k$ is obtained by simple kriging (best linear unbiased estimate of the conditional mean, see section 2.5.3). The kriging weights Λ and the estimated conditional mean $<\mathbf{X_2}>^k$ are retained for the subsequent generation of conditional random fields $\mathbf{X}_s^c$, which are constructed through the following relationship:

$$\mathbf{X}_s^{\mathbf{c}} = <\mathbf{X}>^{\mathbf{k}} + (\mathbf{X}_s - <\mathbf{X}_s>^{\mathbf{k}}) = <\mathbf{X}>^{\mathbf{k}} + \mathbf{e}_w \qquad \textbf{(3-14)}$$

where $<\mathbf{X}>^k$ is the kriged random field given the simulated data $\mathbf{X_{1s}}$ from the unconditionally generated random field $\mathbf{X}_s$. $\mathbf{X}_s$ has a joint probability distribution defined by the measured moments. $<\mathbf{X}_s>^k$ is the simulated equivalent to $<\mathbf{X}>^k$: It preserves the data $\mathbf{X_{1s}}$ in the unconditionally generated random fields at and only at the locations $\{\mathbf{x}_1,...,\mathbf{x}_m\}$, where measurements are available in the real field site as well, and of the kriged estimates $<\mathbf{X_{2s}}>^k$ at all other locations $\{\mathbf{x}_{m+1},...,\mathbf{x}_n\}$ given the unconditionally simulated data $\mathbf{X_{1s}}$. The difference $(\mathbf{X}_s - <\mathbf{X}_s>^k)$ is a realization $\mathbf{e}_s$ of a possible estimation error incurred by estimating the data $\mathbf{X}_s$ through the kriged values $<\mathbf{X}_s>^k$. The simulated error is added to the originally estimated conditional mean $<\mathbf{X}>^k$ to obtain a possible conditional random field $\mathbf{X}_s^c$.

The simulated estimation error $\mathbf{e}_s$ has the same conditional moments as the real estimation error $\mathbf{e} = (\mathbf{X} - <\mathbf{X}>^c)$ because the unconditional pdfs of the real and the simulated fields are identical (neglecting the possibility of measurement and moment estimation errors), and because the conditioning occurs at the exact same locations both at the field site and in the

simulations (Journel, 1974; Delhomme, 1979). Recall from (2-48) that the conditional covariance or error covariance $\mathbf{E}_{22}$ depends only on the location of the conditioning points $\mathbf{x}_1$ and on the unconditional covariance $\mathbf{C}$, but not on the actual value of the conditioning data $\mathbf{X}_1$!

The unconditional random field generation and the kriging of the generated random field from the simulated measurement data are repeated for each realization. Each simulation will yield a random field of estimation errors $\mathbf{e}_s$, which can be added to the kriging estimate of the real data to obtain a conditional random field. For a large number of samples thus obtained, the sample variance of $\mathbf{X}_{2s}(\mathbf{x}_2)$ will converge in the mean square to the true conditional variance or kriging variance of $\mathbf{X}_2(\mathbf{x}_2)$ as shown by Delhomme (1979). It is obvious that this conditioning technique is independent of the method used to generate the unconditional random field and is as such unrelated to the spectral random field generator. The advantages of using this method together with the spectral random field generator will be discussed in chapters 7 and 10.

3.4 Alternative Methods of Random Field Generation

3.4.1 Turning Bands Method

The turning bands method was first proposed by Matheron (1973) to simulate unconditional random fields. Detailed descriptions of the turning bands method can be found elsewhere (e.g. Mantoglou and Wilson, 1982; Brooker, 1985; Mantoglou, 1987; Tompson et al., 1989). For completeness, a brief outline of the structure of the turning band method is given.

The principal advantage of the method is that it reduces the generation of a two- or three-dimensional, random, spatially correlated process to the generation of one-dimensional, correlated line processes. The reduction in dimensionality is made possible by the fact that the transformation from a 3- or 2-dimensional covariance function into an equivalent one-dimensional covariance function can be uniquely defined (Matheron, 1973; Mantoglou and

Wilson, 1982). After determining the equivalent 1-dimensional covariance, a one-dimensional, multivariate process Y(x) can be generated along a finite line by using an appropriate autoregressive or moving average algorithm (Bras and Rodriguez-Iturbe, 1985) or 1-dimensional spectral methods similar to the one described above (Mantoglou, 1987). To obtain the 2-dimensional random field, the one-dimensional simulation is repeated on a total of 16 (or more) equally spaced lines intersecting at their midpoints. Each of these lines is divided into small, discrete intervals of equal size. One random number is generated for each interval. The random value $X(\mathbf{x})$ of a realization at any point $\mathbf{x}$ is computed by averaging the 16 corresponding line values:

$$X(\mathbf{x}) = \frac{1}{\sqrt{16}} \sum_{j=1}^{16} Y(x_j, j) \qquad \textbf{(3-15)}$$

where j is the line number and x_j is located on line j such that $\mathbf{x}$ is orthogonal to x_j with respect to line j.

Conditional simulations with the turning bands method were among the first in hydrologic applications (Delhomme, 1979) and the method used is identical to the one described in section 3.3. for the spectral random field generator, since the actual conditioning is independent of the method used for unconditional random field generation.

3.4.2 Matrix Decomposition

3.4.2.1 Unconditional Simulation by Matrix Decomposition

An elegant approach to simulating unconditional as well as conditional random fields is the matrix decomposition method (Clifton and Neuman, 1982; Davis, 1987; Alabert, 1987). Again, it is assumed that a valid unconditional covariance model is given (satisfying the conditions stated for (2-28)). The covariance between each two points in the random field domain is computed prior to the random field generation and stored in an unconditional

symmetric covariance matrix **C**. For a random field consisting of n points, **C** has a dimension of n^2. Furthermore, it is assumed that the unconditional expected value $\langle X(\mathbf{x})\rangle$ is zero. Using, for example, the Cholesky algorithm for symmetric, positive definite matrices, the covariance matrix can be decomposed into a lower triangular matrix **L** and an upper triangular matrix **U**:

$$\mathbf{C} = \mathbf{L}\,\mathbf{U} \qquad \mathbf{L} = \mathbf{U}^T \text{ (^{T} indicates the transpose operator)} \qquad \textbf{(3-16)}$$

The product of the lower matrix **L** and a vector α of random, uncorrelated, univariate normally distributed random numbers α_i, i=1,...,n with zero mean and unit variance will then give a simulated random field $\mathbf{X}_s$ with the desired mean and covariance:

$$\mathbf{L}\,\alpha = \mathbf{X}_s \qquad \textbf{(3-17)}$$

Proof :

$$\langle \mathbf{X}_s \rangle = \langle \mathbf{L}\,\alpha \rangle = \mathbf{L}\,\langle \alpha \rangle = 0 \qquad \textbf{(3-18)}$$

$$\mathbf{C}_s = \langle \mathbf{X}_s\,\mathbf{X}_s^T \rangle = \langle \mathbf{L}\,\alpha\,(\mathbf{L}\,\alpha)^T \rangle =$$

$$\langle \mathbf{L}\,\alpha\,\alpha^T\,\mathbf{L}^T \rangle = \mathbf{L}\,\mathbf{I}\,\mathbf{U} = \mathbf{L}\,\mathbf{U} = \mathbf{C} \qquad \textbf{(3-19)}$$

(**I** is the identity matrix)

After generating and decomposing the covariance matrix **C** once, any new realization of X is simply obtained by generating a new sample of the random vector α, which can easily be done with any good random number generator. Note that the method is independent of the dimensionality of the random field and that the covariance need not be stationary. Only 1st order (mean) stationarity is required for this method.

3.4.2.2 Conditional Simulation by Matrix Decomposition

The procedure can readily be extended to implement conditional simulations (Clifton and Neuman, 1982; Davis, 1987). Again $\mathbf{X}_1$ is a vector of known data-values and $\mathbf{X}_{2s}$ are the unknown random values that are conditionally simulated. The covariance matrix and its decomposition (3-16) is expanded in the following form:

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_{11} & \mathbf{C}_{12} \\ \mathbf{C}_{21} & \mathbf{C}_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{L}_{11} & \mathbf{0} \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{U}_{11} & \mathbf{U}_{12} \\ \mathbf{0} & \mathbf{U}_{22} \end{bmatrix} \tag{3-20}$$

that is, the following four equations:

$$\mathbf{C}_{11} = \mathbf{L}_{11}\,\mathbf{U}_{11} = \mathbf{L}_{11}\,\mathbf{L}_{11}^{T} \tag{3-21a}$$
$$\mathbf{C}_{12} = \mathbf{L}_{11}\,\mathbf{U}_{12} = \mathbf{L}_{11}\,\mathbf{L}_{21}^{T} \tag{3-21b}$$
$$\mathbf{C}_{21} = \mathbf{L}_{21}\,\mathbf{U}_{11} = \mathbf{L}_{21}\,\mathbf{L}_{11}^{T} \tag{3-21c}$$
$$\mathbf{C}_{22} = \mathbf{L}_{21}\,\mathbf{U}_{12} + \mathbf{L}_{22}\,\mathbf{U}_{22} \tag{3-21d}$$

and as shown for the unconditional simulation (3-17):

$$\begin{bmatrix} \mathbf{L}_{11} & \mathbf{0} \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_{2s}^{c} \end{bmatrix} \tag{3-22}$$

However, $\mathbf{X}_1$ is known and α_1 need not be generated! Instead, the values α_1 are computed by solving the first of the two equations in the matrix (3-22):

$$\alpha_1 = \mathbf{L}_{11}^{-1}\,\mathbf{X}_1 \tag{3-23}$$

where $\mathbf{L}_{11}$ and $\mathbf{X}_1$ are given. α_2 is generated in the same way as α in the unconditional simulation. Then the conditional values $\mathbf{X}_{2s}^{c}$ are computed by solving the second of the the two equations in (3-22):

$$\mathbf{L}_{21}\,\alpha_1 + \mathbf{L}_{22}\,\alpha_2 = \mathbf{X}_{2s}^{c} \tag{3-24}$$

The procedure is further simplified by expressing all terms in (3-24) as functions of $\mathbf{L}_{11}$, the covariance submatrices of $\mathbf{C}$, and the known data array $\mathbf{X}_1$:

$$\mathbf{L}_{21} = \mathbf{C}_{21}\,\mathbf{U}_{11}^{-1} = (\mathbf{U}_{11}^{-1})^{T}\,\mathbf{C}_{12}^{T} = \mathbf{L}_{11}^{-1}\,\mathbf{C}_{12} \tag{3-25}$$

$$\mathbf{L}_{22}\,\mathbf{U}_{22} = \mathbf{C}_{22} - \mathbf{L}_{21}\,\mathbf{U}_{12} = \mathbf{C}_{22} - \mathbf{L}_{11}^{-1}\,\mathbf{C}_{12}\,(\mathbf{L}_{11}^{-1}\,\mathbf{C}_{12})^{T} \tag{3-26}$$

Collecting (3-23), (3-25), and (3-26) in (3-24), the conditional simulation becomes:

$$\mathbf{X}_{2s} = \mathbf{L}_{11}^{-1}\, \mathbf{C}_{12}\, \mathbf{L}_{11}^{-1}\, \mathbf{X}_1 + \mathbf{L}_{22}\, \alpha_2 \qquad \textbf{(3-27)}$$

Thus, the initial steps to conditional simulation are:

1. decompose $\mathbf{C}_{11}$,
2. invert the resulting matrix $\mathbf{L}_{11}$, and
3. decompose (3-26) in order to obtain $\mathbf{L}_{22}$.

Once all these matrices are determined, new realizations of conditional $\mathbf{X}_{2s}^{c}$ are obtained by simply generating new α_2 with a Gaussian random number generator and solving (3-27). It can be shown that the moments of $\mathbf{X}_{2s}^{c}$ are exactly the conditional moments defined in (2-46) and (2-48) (Alabert, 1987; Harter, 1992).

As an alternative to the above approach, Clifton and Neuman (1982) suggested to obtain the kriging estimate $\mathbf{X}_2^{k}$ of the points to be simulated. Then an error $\mathbf{e}_s$ with covariance $\mathbf{E}_{22} = (\mathbf{L}_{22}\, \mathbf{U}_{22})$ is generated taking advantage of the matrix decomposition method introduced in (3-17):

$$\mathbf{L}_{22}\, \alpha_2 = \mathbf{e}_s \qquad \textbf{(3-28)}$$

Although based on the same theoretical foundation as the suggestion by Alabert (1987), just one matrix ($\mathbf{E}_{22}$) needs to be decomposed instead of 2 as outlined in (3-27). However, $\mathbf{C}_{11}$ must be inverted to obtain the kriging weight matrix Λ_{12} and the kriging estimates $\mathbf{X}_2^{k}$ (2-46). (3-28) is very general in that its application is not limited to simple kriging. Ordinary or universal kriging estimates with the ordinary or universal kriging covariances can be applied as well as others e.g., Bayesian estimates for inverse modeling (the latter was implemented by Clifton and Neuman, 1982).

3.4.3 Gaussian Sequential Simulation

Sequential simulation was first implemented by Journel and Gomez-Hernandez (1989). Their version of the sequential simulator was specifically designed to generate stationary random fields with a non-parametric probability distribution ("indicator simulation"). Gomez-Hernandez (1991) presented a sequential simulator for the generation of multivariate normal random fields. The Gaussian simulation technique goes back to the definition of unconditional and conditional probabilities (2-42) and (2-43): Rearranging equation (2-43), the joint probability $f_X(x_1,x_2,...,x_n)$ is expressed as a function of the marginal distribution of the known data $x_1,....x_m$ and the conditional distribution of the unknown RVs $X_{m+1},..., X_n$:

$$f_X(x_1,x_2,..x_m,x_{m+1},...,x_n) = f_X(x_{m+1},xm+2,...,x_n|x_1,x_2,...,x_m)\ f_X(x_1,x_2,...,x_m) \quad \textbf{(3-29)}$$

The equation is expanded into a sequence of lower order conditional probability terms:

$$f_X(x_1,x_2,...x_m,x_{m+1},...,x_n) =$$
$$f_X(x_n|x_1,x_2,....,x_{n-1})\ f_X(x_{n-1}|x_1,x_2,....,x_{n-2})\ f_X(x_{n-2}|x1,x2,....,x_{n-3}) \quad \textbf{(3-30)}$$
$$.....f_X(x_{m+1}|x_1,x_2,....,x_m)\ f_X(x_1,x_2,...,x_m)$$

It is this form of the joint probability density function, which gives rise to the sequential simulator: The conditional probability function of each $f_X(x_{m+i}|x_1,...,x_m,x_{m+1},...,x_{i-1})$, $m < m+i \leq n$, can be expressed as a product of (i-1) **univariate** conditional density functions and the unconditional density function of the known data $x_1,...,x_m$. Hence the simulation algorithm for a realization is the following:

1. select a datapoint x_{m+1}^c to be generated,
2. find the conditional density for that datapoint given the measured data $x_1,...,x_m$ and draw

from it x_{m+1}^c,

3. select a second datapoint x_{m+2}^c to be generated,
4. find the conditional density for that datapoint given the measured data $x_1,...,x_m$ **and** given the already generated datapoint x_{m+1}^c and draw from it x_{m+2}^c,
5. repeat the procedure until a conditional sample has been drawn for all points.

For each realization, steps 1.-5. are implemented independently. The procedure as such is entirely general and can be applied to any random process.

The procedure is again illustrated for the multivariate normal, stationary, zero-mean random process. Specifically it is shown how 'to draw [a random number x_{m+i}] from [a univariate conditional] density function' (Gomez-Hernandez, 1991, p.42). First, the moments of the univariate conditional density function $f_X(x_{m+i}|x_1,...,x_m,x_{m+1},...,x_{m+i-1})$, $m < m+i \leq n$, must be specified. By definitions (2-46) and (2-48), the conditional mean and covariance are given by the kriging estimate x_{m+i}^k and the kriging covariance E_{ij}. The kriging estimate x_{m+i}^k is computed from **both** measured $x_1,...,x_m$ **and** already generated data $x_{m+1}^c,...,x_{m+i-1}^c$:

$$X_{m+i}^k = \Sigma \lambda_{ij} X_i + \Sigma \Lambda_{il} X_l \qquad j = 1,...,m; \quad l = m+1,...,m+i- \tag{3-31}$$

As the conditional density is univariate, only the kriging variance E_{ii} (2-48) is relevant:

$$E_{ii} = \sigma^2_X - \Sigma C_{il} \lambda_{li} \qquad k = 1,...,m,...,m+i-1 \tag{3-32}$$

The important difference between this and all previously described methods is that a **univariate** conditional random variable is generated, which by definition renders the consideration of the error **co**variance i.e., the spatial correlation structure of the error, superfluous. $X_{(m+i)s}^c$ is simulated by first obtaining the kriging estimate X_{m+i}^k and then adding a random error $e_{(m+i)s}$, which is drawn from a zero-mean, univariate normal distribution with variance E_{ii} (3-32):

$$X_{(m+i)s} = X_{m+i}^k + e_{(m+i)s} \tag{3-33}$$

No covariance matrices are involved in the generation of the error $e_{(m+i)s}$ thus reducing the task of random field generation to generating independent, univariate random numbers.

Note that the procedure can be equally applied to unconditional simulation and conditional simulation. In unconditional simulations, the first point is generated as an independent random variable with the desired unconditional mean and variance. A second datapoint is generated conditioned on the first one in the manner described by (3-33), and so on. In a conditional simulation the generation of random fields begins with the already measured data. Subsequent datapoints are generated as in an unconditional simulation through the conditional relationship (3-33). For further details of the method, see Gomez-Hernandez (1991).

3.5 Performance Analysis of the SRFFT, LU, TB, and S Random Field Generators

3.5.1 Design of the Performance Analysis

Random field generators must generate truly independent realizations $X(\mathbf{x})$ of an RFV such that the sample joint probability distribution of the random field realizations converges in mean square to the desired ensemble probability distribution (the pdf of the assumed probability space) in the limit as the number of realizations becomes infinite. Generally, two conditions must be fulfilled for a random field generator to give statistical results that converge in the mean to the desired ensemble moments of the RFV:

1. the random number generator (RNG) must be able to generate independent normally distributed random numbers with given mean μ and variance σ^2 in the limit as the sample size becomes large.
2. the numerical implementation of the random field generator (RFG) must be free of deterministic artificial patterns caused by the generating algorithm.

The first condition can be tested separately and should be tested prior to a performance

analysis of a random field generator. Random number generators are common in many scientific programming libraries. Two random number generators are tested both of which produce independent random samples distributed uniformly between 0 and 1: the random number generator SURAND in the Engineering and Scientific Subroutine Library (ESSL) available with the Fortran compilers for IBM workstations (IBM, 1993) and the random number generator program RAN2 described by Press et al. (1992). The normally distributed random numbers are obtained after a transformation achieved through the subroutines SNRAND (IBM, 1993) and GAUSDEV (Press et al., 1992) both of which use the Box-Muller method to obtain a normal deviate from a uniform deviate (Knuth, 1981).

Random number generators are not truly random. They rely on a deterministic formula to generate a new random number. Both random number generators tested are based on a congruential algorithm which algebraically alters a given number. Initially this number is directly or indirectly supplied by the user. In subsequent generations within the same program execution the number is taken from the preceding generation of a random number (Knuth, 1981). Sparing the details of the algorithms, note that one of the most important properties of good random number generators is the independence of subsequently generated random numbers and the time to recurrence i.e., the number of random numbers generated before any previously generated random number is generated a second time. Once the seed to a random number is regenerated the second time, the following sequence of random numbers will be exactly the same as the sequence of random numbers generated after the first occurrence and hence will repeat itself *ad infinitum*. Since all computers have finite accuracy, they can only generate a finite number of different discrete numbers. Thus, every random number generator will eventually generate a random number that has already occurred before. The length of the random number sequence to the first recurrence is desired to be large, much larger than the

number of samples actually generated in an application to assure the independence of the samples. Theoretically, infinite sequences can be achieved by appropriately shuffling small parts of the sequence (Press et al., 1992). For SURAND and RAN2 no recurrence of the initial seed was found within the following 10^{12} numbers, after which the test was interrupted.

A number of methods exist to test random number generators (Knuth, 1981). Sharp and Bays (1992) test the independence of consecutive random numbers by plotting the two-dimensional coordinates given by any two consecutive random numbers as dots into a map (see Orr, 1993). Biasedness can then often be discovered (although not always) if certain patterns develop. Unbiased uniform random number generators should fill such a map evenly. Figure 3.1 shows a sample of 20000 pairs generated with SURAND and RAN2. There is no obvious artificial pattern in the samples and - at least qualitatively - they are indeed from a uniform distribution. The CPU-times of both random number generators are comparable. In cooperation with Orr (see Orr, 1993) three other congruential random number generators were tested with similar results: DPRAND by Maclaren (1992), which is a portable random number generator like RAN2, and the random number generators in the IMSL library (1991) and NAG library (1990).

All five random number generators are tested by the author within the spectral random field generator (original FORTRAN code provided by Gutjahr, 1989) and within the LU-decomposition based random field generator (FORTRAN code developed by the author) and as shown below, no specific bias was detectable due to the random number generator.

In this study the spectral random field generator is used for reasons that will become obvious in chapter 7. To assure its proper performance a large Monte Carlo simulation with 1000 samples was performed on a square grid with 64^2 gridpoints. The mean μ and variance σ^2 are 0 and 1, respectively. The covariance is isotropic and exponential (Isaacs and Srivastava,

1990):

$$C(\xi) = \sigma^2 \exp(-\frac{|\xi|}{\lambda}) \qquad \textbf{(3-34)}$$

where λ is a parameter called the "integral scale" of the covariance function and is here specified to be five gridpoint increments, a commonly chosen discretization of the random field. Commonly, the integral scale is referred to as the "correlation length" or "correlation scale" of C. For comparison, an equivalent Monte Carlo simulation is performed with a LU-decomposition based random field generator. Both the SRFFT and the LU Monte Carlo simulation are performed once with each of the five random number generators. Orr (1993) provided test-results from identically implemented TB and S Monte Carlo simulations using the DPRAND random number generator. The turning bands method is based on FORTRAN code by Zimmermann and Wilson, 1990. The sequential simulator in C code has been provided by Gomez-Hernandez (1991).

For the evaluation of the Monte Carlo simulations, a postprocessor was developed that is designed to collect the following sample moments:

* (spatial) mean and variance of each realization by summing over all values in a realization.
* (spatial) sample covariance of each realization by inverting the procedure of the spectral random field generator: take a Fourier transform of the realization $X(\mathbf{x})$ to obtain the spectral representation $dZ(\mathbf{k})$, take the expected value of the spectral representation with its conjugate to obtain the spectral density function, and compute the spatial sample covariance from the inverse Fourier transform of the sample spectral density function (Gutjahr, 1989).
* (local) mean and variance at location l as a function of the number of realizations (10,

20, 50, 100, 200, 500, and 1000 samples).

* (local) sample covariance at point l by performing the following summation over all N realizations for N=10, 20, 50, 100, 200, 500, and 1000 samples:

$$Cov_l(k) = \frac{1}{N}\sum_{n=1}^{N} X_l^n X_k^n - \frac{1}{N}\sum_{n=1}^{N} X_l^n * \frac{1}{N}\sum_{n=1}^{N} X_k^n \qquad \textbf{(3-35)}$$

where here n is an index to X and not a power of X. The points k surrounding the point l consist of all the points in a squared window of side-length 32 (33 points) centered on point l. This summation was implemented only for the 31 by 31 points in the center of the 64^2 grid. This sampling pattern avoids problems with boundaries and allows the evaluation of the local sample covariance on a complete squared window.

The sample moments are further evaluated statistically to give several summary moments:

* the average of the spatial means of each realization (which is exactly equal to the spatial average of the local means) gives the total mean of all numbers generated in the Monte Carlo simulation and must converge in the mean square to zero as the number of realizations (samples) increases.

* the average of the spatial variances is expected to be smaller or equal to the local variances (due to spatial correlation), and the spatial average of the sample local variances must converge in the mean square to the desired local ensemble variance as the number of runs in the Monte Carlo simulation increases.

* the average of the local sample covariances (over all locations) must converge in the mean square to the desired exponential covariance function. Like the average spatial variance, the average of the spatial sample covariances is expected to be smaller or equal to the specified covariance. The average spatial covariance normalized by the average spatial variance, however, must be equal to the specified (normalized)

exponential covariance function. The deviation of the average local covariance function from the specified exponential covariance function is computed. The local deviations are integrated over the entire domain of the two-dimensional covariance to obtain a mean deviation from theoretical covariance. Given the set of 31^2 local covariances, each of which is a 33^2 point two-dimensional field, the variance of the sample local covariance is calculated as a function of the separation point k and integrated over all k to obtain the average variance of the local covariance.

* the minimum and maximum of all local or spatial moments give a range of possible sample moments. The range is expected to decrease as the number of samples in the Monte Carlo simulation increases.

3.5.2 Summary Performance of the RNGs and RFGs

First, the summary moments are evaluated from the Monte Carlo simulations with the spectral (SRFFT) and the LU-decomposition (LU) random field generators (RFGs) using the five different random number generators (RNGs) mentioned above.

Figure (3.2a) shows the total sample mean as a function of the number of SRFTT Monte Carlo realizations (NMC) for each of the five random number generators. The differences in sample means shown for the different RNGs reflects the sample moment variability, since each RNG generates a different sequence of random realizations. Initially, the differences are large due to the limited number of samples. Note that although there are $64^2 = 4096$ samples of random numbers within each realization, these 4096 samples are not independent of each other and the spatial sample mean does not converge to the ensemble mean as fast as that of NMC * 4096 uncorrelated random samples. With all five random number generators the total sample

mean (spatial mean of the local sample means) converges to zero as the number of realizations increases. A very similar behavior is seen in the total mean of the LU generator (Figure 3.3a). The convergence rate of the different RNG-based Monte Carlo simulations is the same for both random field generators: after 10 realizations the total sample mean varies within the range [-0.1, 0.1]; after 100 realizations the range decreases to [-0.025, 0.025]; and after 1000 realizations it reaches the limits [-0.01, 0.01]. No particular bias (i.e. numerical artifact) is found in any of the five RNGs with any of the two RFGs regarding the total sample mean. The decrease in the range of the sample mean is consistent with the theoretical decrease of the variance of the sample mean as a function of the sample size (Haan, 1977; see chapter 8).

Both the average of local sample variances and the average of the spatial sample variances converge (in the mean square sense) to the ensemble variance specified as the number of realizations becomes large (Figures 3.2c, 3.3c). Due to the spatial correlation the mean spatial variance is approximately 5% lower than the mean local variance with both the SRFFT and LU generators. Unexpectedly, however, it is found that the mean local variance for all five RNGs in the SRFFT generator converges to values between 0.94 and 0.95, which is approximately 5% below the specified unit variance. The mean local variance of the LU generator converges to 1 for all five RNGs. Hence the erroneous sample variance in the SRFFT simulations are solely due to the procedure in the SRFFT random field generation and not due to the random number generators used. This is an important drawback of the SRFFT generator, which is addressed in more detail below.

The range of local variances for the SRFFT and the LU random field generators are comparable (Figures 3.2b, 3.3b and independent of the RNG used: After 10 realizations, local variances vary approximately between [0.1, 4], after 100 realizations the range is limited to [0.5, 1.8], and after 1000 realizations the range is approximately [0.7, 1.3]. The decrease in

variability is due to the greater number of samples from which each local variance is computed. On the other hand the number of samples from which the spatial variance is computed is always 4096, and with each realization a sample of the statistics "spatial variance" is added. The range of these samples therefore slightly increases as more realizations are added (Figures 3.2d and 3.3d). At 1000 realizations the spatial variances vary approximately between [0.6, 1.7] for both the SRFFT and the LU generator. Again, the minimum and maximum spatial variance of the SRFFT random fields are approximately 5% lower than those of the LU simulations. For all variance computations none of the RNGs produces results significantly different from others.

The total average deviation of the sample local covariance from the specified exponential covariance (averaged over all points k in the covariance field l AND averaged over all covariance realizations l) varies around 0 and converges to near 0 as the number of realizations increases. Results for the SRFFT and the LU simulations are very similar, independent of the RNG generator chosen (Figures 3.2e and 3.3e). This shows that the deviation of the variance in the SRFFT simulations from the specified variance does not occur for the entire sample covariance. Indeed, a comparison of the mean local sample covariance obtained from a SRFFT simulation with the exponential covariance specified shows that the erroneous deviation of the SRFFT sample covariance function is limited to the center (origin) of the covariance function i.e., to the variance itself (see below).

The average variance, and the maximum, and minimum variance of the sample covariance function, averaged over all points in the sample covariance, decrease such that their logarithms (the logarithm of the mean, the minimum, and the maximum) decrease linearly with the logarithm of the number of Monte Carlo realizations (Figures 3.2f and 3.3f), which is in good agreement with the statistical analysis: the standard deviation of the sample moments of independent random variables theoretically decreases proportional to $1/n^{1/2}$ (c.f. Haan, 1977).

The results for the SRFFT and the LU simulations are again nearly identical and independent of the random number generators used.

Using the postprocessing program developed by this author, Orr (1993) also computed summary statistics for the turning bands (TB) and the sequential simulator (S), with which he had implemented Monte Carlo simulations under the same conditions as the above described SRFFT and LU simulations. Only the DPRAND subroutine was used as random number generator (Maclaren 1992). A comparison of the summary statistics of the four Monte Carlo simulations with the DPRAND random number generator and the SRFFT, the LU, the TB, and S random field generators are shown in Figures 3.4a through 3.4f. In general both the TB and S simulations give results that are - for all practical purposes - identical to the SRFFT and LU simulations. Neither the TB nor the S simulations show any bias regarding the mean local variance, which is generally too low for the SRFFT simulations (Figure 3.4c). The only notable exception is a relatively high average deviation of the sample covariance from the specified covariance in the S simulations (Figure 3e): After 1000 realizations the mean deviation in the S simulation is approximately five times higher than in any of the other RFG simulations, and approximately 2.5 times larger than in any of the simulations with the other four RNGs in the SRFFT and LU RFGs (compare Figure 3.4e with Figures 3.2e and 3.3e). This may indicate that the Sequential Simulator produces a slightly higher sample covariance than specified, either overall, or in a small region within the covariance field.

In conclusion, the summary statistics indicate that any of the five random number generators tested will produce reliable results. With respect to the summary statistics, all four random field generators produce results that converge in the mean square sense to the desired ensemble distribution when the number of realizations is large. The two exceptions are: First, the covariance of the SRFFT simulations is significantly lower (about 5%) at and only at the

origin of the covariance field. In other words, the variance is too low while all covariances between two different points are statistically accurate. Second, the S simulations may produce a seemingly significant overall deviation in the covariance field such that the sample covariances are on average larger than the specified covariance. The summary statistics have also shown that unless the number of Monte Carlo realizations exceed several hundred or even a thousand runs, local statistics (such as the local mean, variance, and the local covariance field) have a very wide spread and their local statistical significance is questionable.

3.5.3 Local Performance Analysis of the RFGs

The purpose of an analysis of the local moments is to investigate possible spatial bias in the realizations due to the particular random field generator. With the experiences from the above summary analysis it is sufficient to limit the analysis of the two-dimensional datasets to Monte Carlo simulations with N = 1000 realizations. First a single sample realization of each RFG simulation is presented together with its spatial covariance field. Then the local mean, the local variance, the local covariance, and finally the average of the local covariances, and the average of the spatial covariances are analyzed. The author gratefully acknowledges the work by Orr (1993), who implemented the TB and S simulations.

Figure 3.5 shows a representative single realization of each RFG simulation. No particularly disconcerting features are observed. Similar observations were made for other realizations, and generally found no particular notable bias within any one realization. Their spatial covariances (obtained by spectral analysis as described above) are generally more or less symmetrical and exponential near the very center (the origin of the covariance field) but also characterize some of the strong spatial features in the particular random realization e.g., the east-west trending valleys in the particular TB realization (Figure 3.6). It must be emphasized that any realization generated with any one of the RFGs may produce more or less dominant features that are then reflected in the sample covariance function (due to the limited field size). The summary statistical analysis has shown, however, that *overall* these features are well within the theoretically possible sample space. The following analysis will investigate, whether any *local* artifacts exists that are due to the numerical algorithms.

The local mean of 1000 realizations with each RFG are shown in Figure 3.7. The standard deviation of the local sample means e.g., in the LU simulation, is 0.0329. This

compares well with the theoretical standard deviation of the sample mean taken from a Gaussian distribution, which is 0.0316 for a sample-size of 1000 (c.f. Haan, 1977). In accordance with stochastic theory the sample means are a random field variable themselves. The correlation structure of the sample mean field is not quite unlike that of the underlying random fields, something that is observed throughout the remainder of this study. The sample mean field (actually a realization of the sample mean RFV) shows no particularly strong trend or non-stationarity or other artificial patterns.

Similarly, the local variances of 1000 realizations are themselves a random field realization with a familiar looking random pattern that reflects the fact that the correlation structure of the variance field is - like that of the mean field - similar to the correlation structure of the underlying random fields (Figure 3.8). Again no particular trend, non-stationarity, or pattern is observed that may be an artifact of the particular random field generator. The only exception is a very notable streak-line structure in the variance field of the TB simulation. From the left lower origin four lines extend radially throughout the variance field, dividing it into five equally sized pieces (with the exception of the leftmost and the lowest piece, which are only about half the size of the three others). The lines reflect four of the 16 turning bands used for the generation of the random field and are characterized by higher than normal variance on the counterclockwise side and a lower than normal variance on the clockwise side along the imaginary line. These patterns have been reported elsewhere and can be partly eliminated by increasing the number of turning bands. Orr (1993) implemented an alternative TB algorithm provided by Zimmerman (personal communication) and indeed found no artificial patterns in the sample variance (random) field (Figure 3.15) when using 32 lines.

A sample local covariance field (actually also a realization of the sample local covariance RFV) centered around the (48,48) coordinate of the random field is shown in Figure

3.9 for each of the four RFGs. Despite the relatively large number of realizations (many hydrogeologic and soil physical applications of the Monte Carlo method are limited to a few tens or a few hundred simulations), the local covariance functions exhibit a significant amount of randomness. More importantly perhaps they show anisotropy and other irregular structures with all of the four RFGs. This is expected since the statistical significance of 1000 independent samples of the covariance product sum (2-29) is relatively weak. Recall that a similar variability is observed for the local sample variance, which has a sample range of ± 0.2 (or ± 20% of the specified standard deviation).

To obtain a larger sample base, all 31^2 local covariance fields such as those in Figure 3.9 are averaged to obtain a mean local covariance field. The 31^2 local covariance fields are not statistically independent due to the correlation structure of the random field. Nevertheless the mean local covariance field has a very regular structure (Figure 3.10) since the sample error is now much s maller than the range [0,1] of the underlying covariance function. The shapes of three of the four mean local covariance fields is very similar to the specified isotropic exponential covariance function. The TB simulator generates anisotropic random fields with the correct variance, but longer correlation than specified in the horizontal direction and shorter correlation than specified in the vertical direction. Furthermore, a strong lineation is visible, when the difference is plotted between the mean sample local covariance of the TB simulation and the exponential covariance (Figure 3.11). As discussed above for the local variance, this artificial TB pattern is believed to be due to the small number of turning bands chosen for the simulation. Orr (1993) reports that these patterns vanish when a much larger number of turning bands is used with an improved version of the TB generator program. Careful visualization reveals that the mean covariance field of the improved program is indeed very accurate, but the lineations in the deviation from the exponential covariance still exist, albeit at a much smaller

amplitude than before (Figure 3.15). The SRFFT simulator has a significant deviation only at the origin of the covariance field i.e., the variance is biased, while all covariances of non-zero lag seem to converge to the specified structure (Figure 3.11). After correspondence with the author of the original SRFFT generator, Allan Gutjahr, it is not entirely clear what causes this particular bias. A larger size of the random field domain reduces the error. Similarly, the mean covariance field of the sequential simulator is somewhat more stretched out than expected, which possibly explains the positive bias in the total deviation from the specified covariance (see summary statistics discussion). Overall, the LU generator gives the most unbiased mean local sample covariance field.

Figure 3.12 depicts the local variance of the sample covariances corresponding to the local mean of the 31^2 superpositioned local covariance fields in Figure 3.10. All generators have the largest variance near the origin due to the large absolute value of the mean covariance field near and at the origin. Overall the LU generator exhibits the smallest variance. The S simulator exhibits relatively large variances throughout a large central part of the covariance field. Both the S and TB generators also exhibit areas of large variance near the edges of the covariance field, where the absolute value of the covariance is near 0. Again not too much significance should be given to these patterns without sampling from much larger populations i.e., without analyzing a Monte Carlo simulation based on a sample size several orders of magnitude larger.

Finally, the average of the first 50 spatial covariance field samples are analyzed, each of which was obtained from a spectral analysis of a single random field realization (Figure 3.13). The SRFFT, S, and LU generators have average spatial covariance fields very similar to the average local covariance fields in Figure 3.10. The TB generator, however, exhibits both strong anisotropy and the familiar starlike pattern in its mean spatial covariance. The deviation of the

mean spatial covariance from the specified exponential covariance (Figure 3.14) clearly shows the location of the 16 turning bands in the TB generator. Again, the same bias can qualitatively be observed in the improved version of the TB generator (Figure 3.15).

3.6 Conclusion

In summary of the moment analysis it is found that all random number generators tested perform equally well. Of the four random field generators tested, the LU-decomposition based simulation showed the least artificial bias. The local moment analysis confirmed that the SRFFT generator produces sample covariances that are very close to the specified covariance with the exception of the variance (covariance of zero-lag), which is on average about 5% too low. The error is probably due to the limited domain size. The sequential simulator produces sample covariances that are on average slightly larger than the specified covariance throughout most of the sample covariance field. Otherwise both the SRFFT and the S simulator produce random fields that are consistent with the probability space specified. The second order moments of the turning band simulator exhibited significant artificial patterns due to the starlike distribution of the 16 turning bands used. An improved code and the choice of a larger number of turning bands gave results comparable to those found for the other three RFGs (see Orr, 1993, for details).

The CPU-efficiency of the four RFGs varies greatly, while the choice of the RNG has no significant effect on the computation time. The CPU time for 1000 realizations of the SRFFT and the LU simulators were 1112 sec. and 1920 sec., respectively, on an IBM RS6000/320 system. The computation time of the SRFFT is proportional to the number of random fields that are generated. In contrast, the LU simulator initially requires large amounts

of computation time just for the decomposition of the covariance matrix. The CPU time for 10 realizations of the LU simulator is 360 sec. compared to only 29 sec. for 10 realizations of the SRFFT simulator. Unfortunately, no CPU-times are available for the simulations with the TB and S simulators by Orr (1993). Tompson et al. (1989) evaluated the computing efficiency of the TB method as compared to the SRFFT method and concluded that the SRFFT will be at least as efficient as the TB method for random fields on the order of less than 10^5 points. For very large random fields, the TB simulator is more efficient, and Gomez-Hernandez (1989) claims similar efficiency for the sequential simulator. The comparative efficiency of the four different RFGs will mainly depend on the number of points generated in each field and on the number of realizations.

While the LU generator gives very good results, its disadvantage is that it requires the decomposition of a covariance matrix of size N^2, where N is the total number of points in the random field (4096 in the above examples). For smaller random fields (<5000 points) with many realizations this is indeed a very effective way of random field generation, since the covariance matrix must only be decomposed once for an entire simulation. Each realization then simply requires the generation of random numbers and the multiplication with the L matrix (3-17). For large random fields, the LU-decomposition becomes too cumbersome, if not impossible.

The largest drawback of the SRFFT generator is the overhead in the FFT, since the actual random field is only $(1/2)^2$ (in two dimensions) or $(1/2)^3$ (in three dimensions) of the size of the spectral field due to the symmetry (3-2) required to obtain real random fields. Gutjahr (1989) points out that the imaginary part of the inverse Fourier transform of the spectral representation is also an independent realization with the required properties i.e., one transform generates two independent realizations of the same random field. For our purposes the SRFFT

is sufficiently accurate (in the statistical sense) and CPU-efficient to justify its use for the simulation of heterogeneous soils.

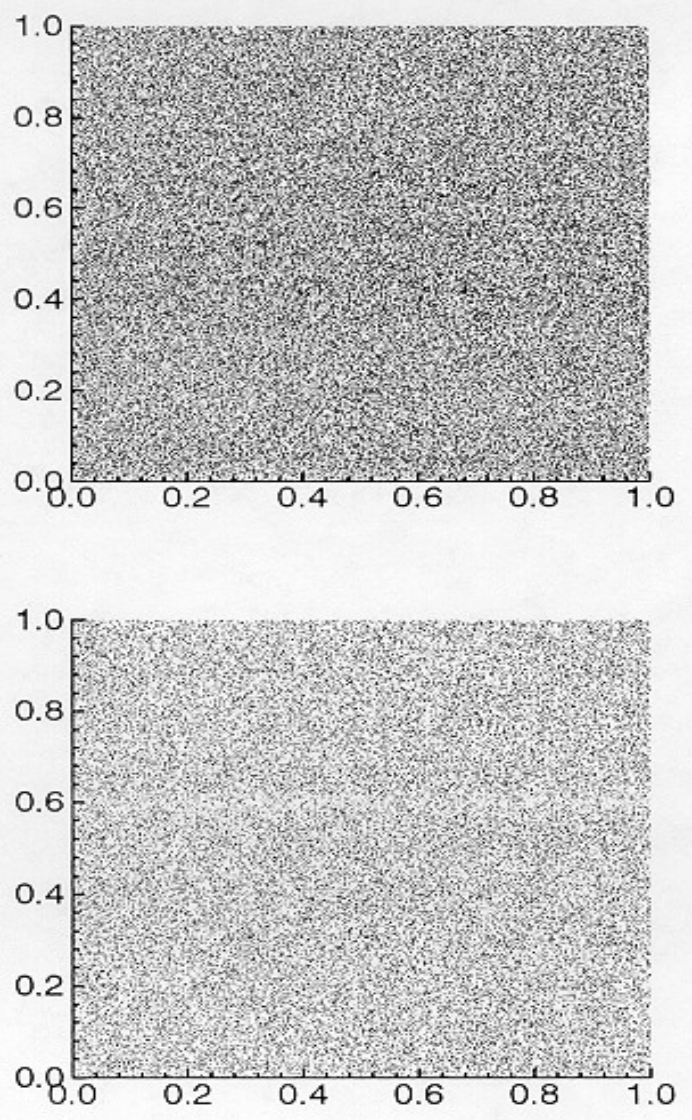

Figure 3.1: Dot diagrams of the RAN2 (top) and ESSL (bottom) random number generators. Each of the 100,000 points represents two consecutive, uniformly distributed random numbers.

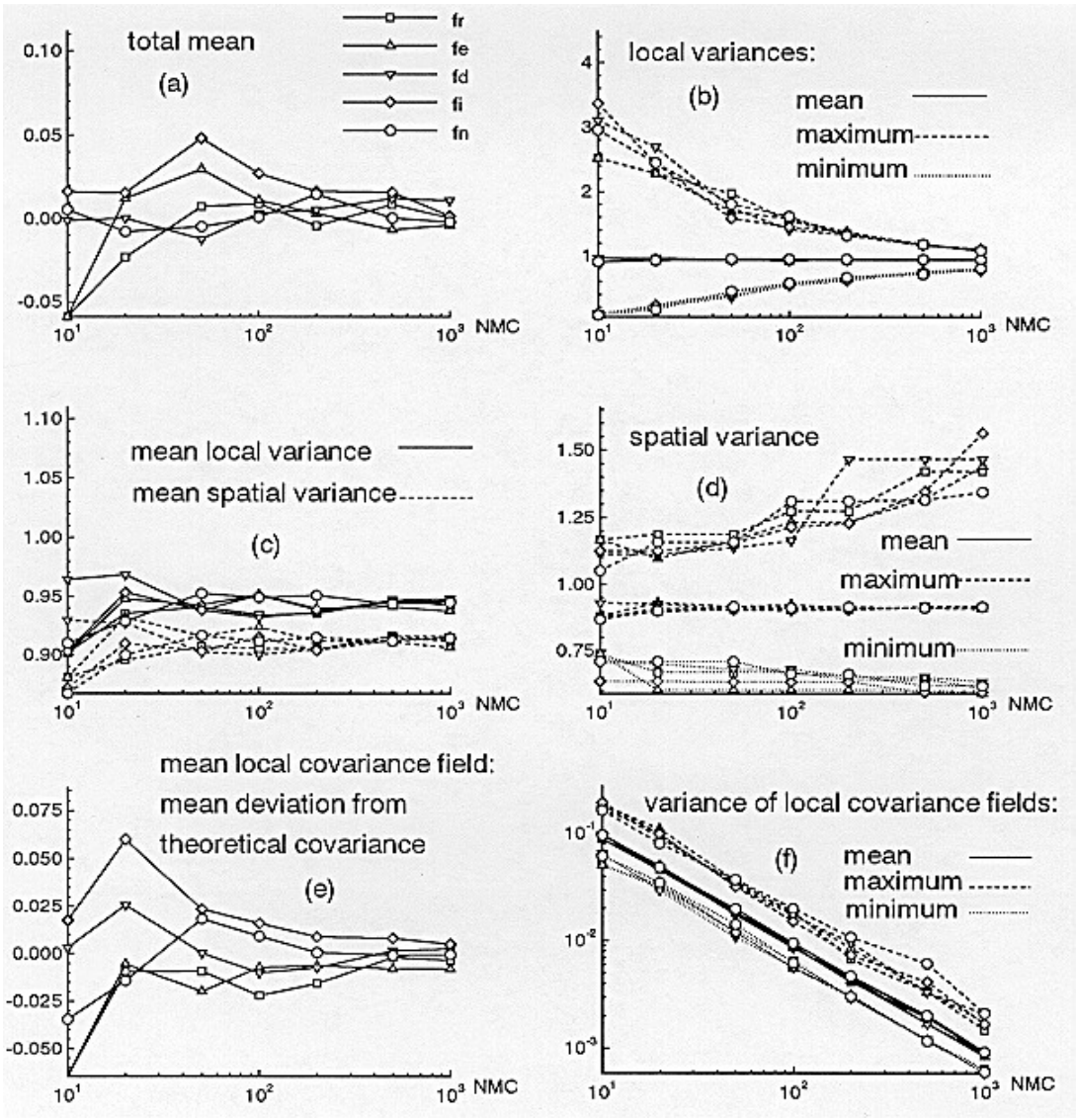

Figure 3.2: Summary moments of the sample mean, variance, and covariance as a function of the number of Monte Carlo realizations, NMC. Local samples moments are taken at the same point over all realizations. Spatial samples are taken from a single realization by sampling over all points. All simulations are implemented with the SRFFT simulator. Different symbols refer to dfiferent random number generators: fr - RAN2, fe ESSL library, fd DPRAND, fi IMSL library, fn NAG library. The f in the labeling refers to SRFFT.

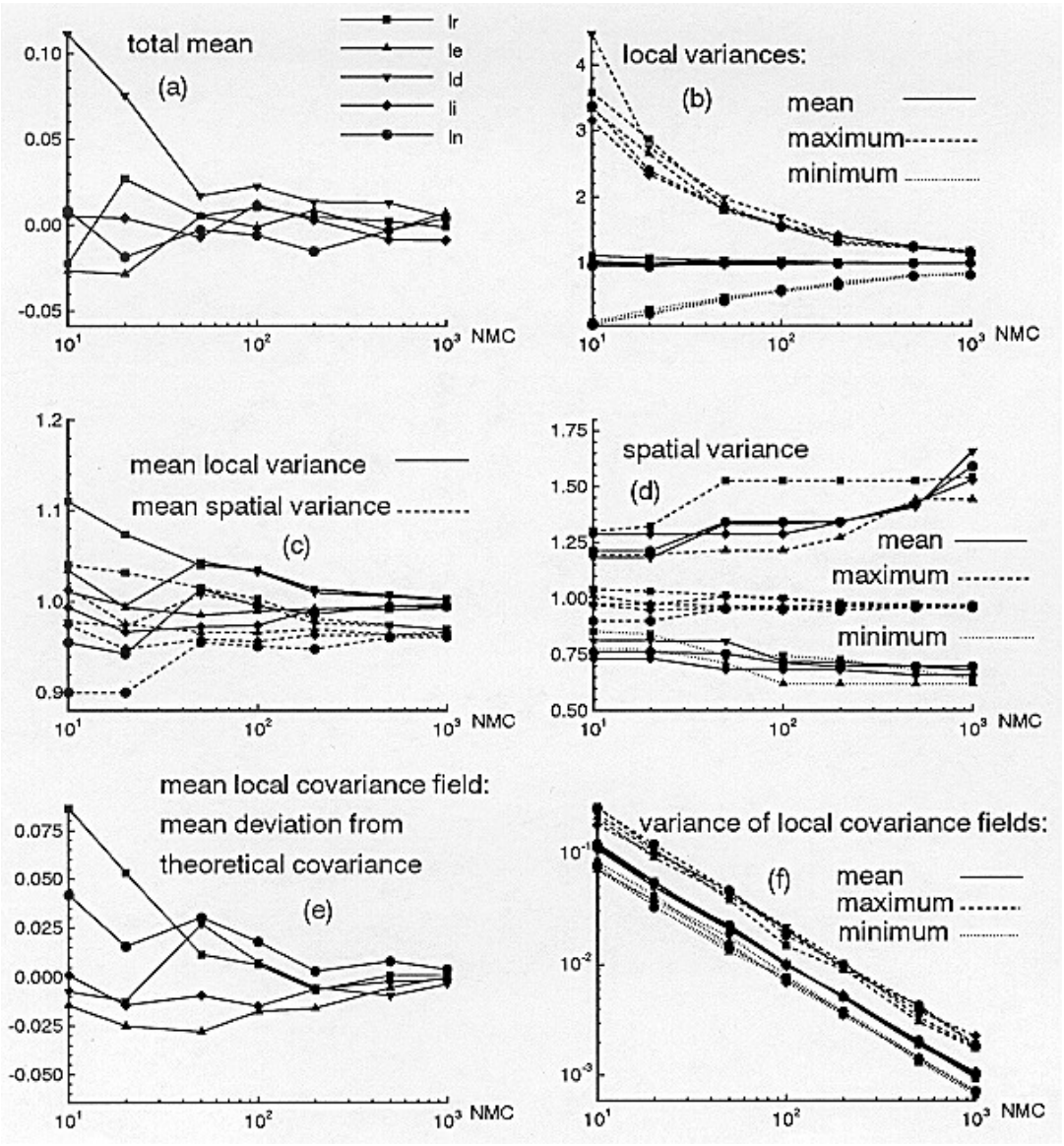

Figure 3.3: Summary moments of the sample mean, variance, and covariance as a function of the number of Monte Carlo realizations, NMC. The simulation results shown here are from simulations with the LU random field generator. The I in the labeling stands for LU random field generator", otherwise the labeling is identical to Figure 3.2.

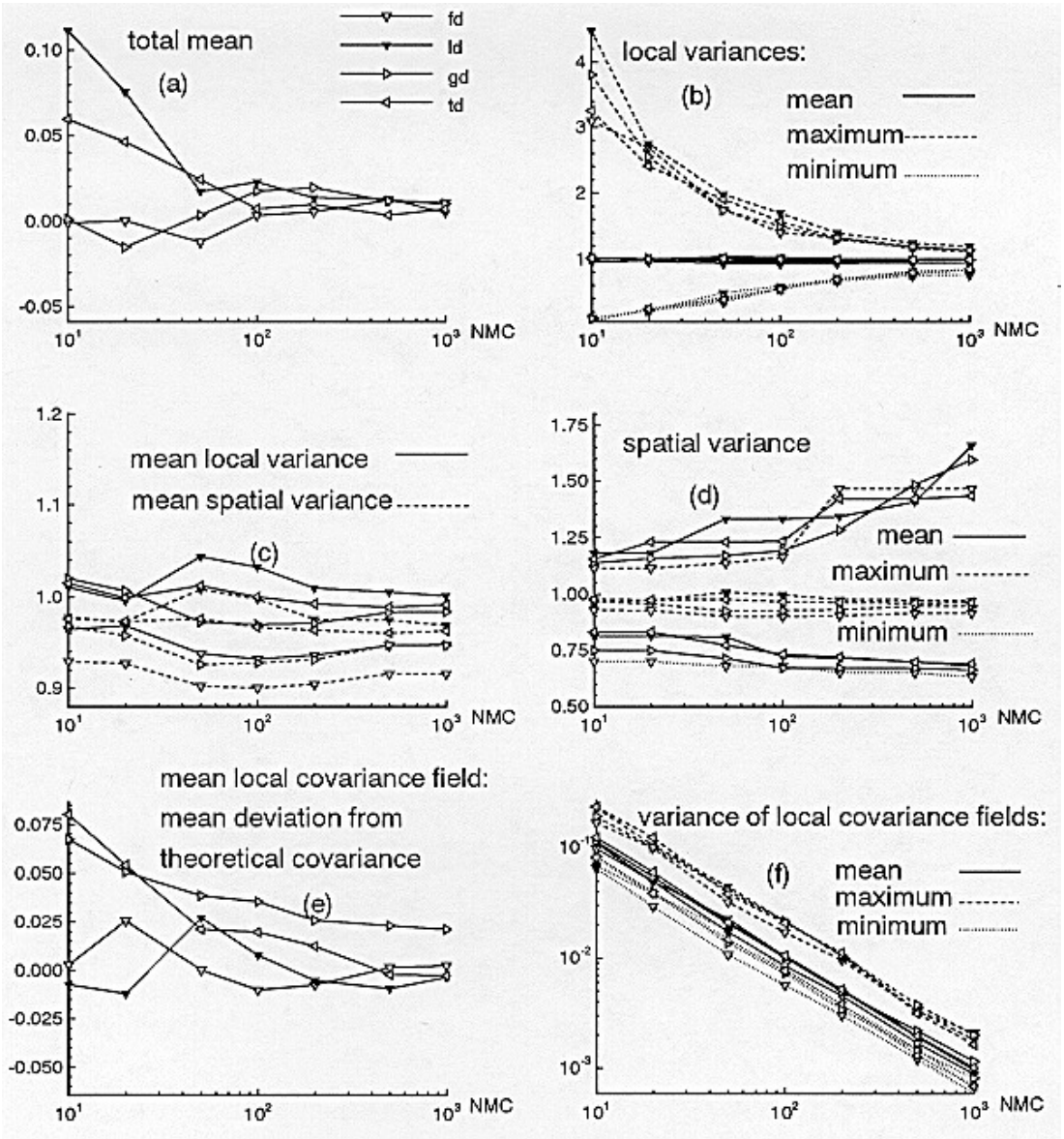

Figure 3.4: Same as Figures 3.2 and 3.3. Here the four different RFGs are compared using the same random number generator (DPRAND). "g" and "t" are the labels for the GCOSIM and the TB simulator, respectively.

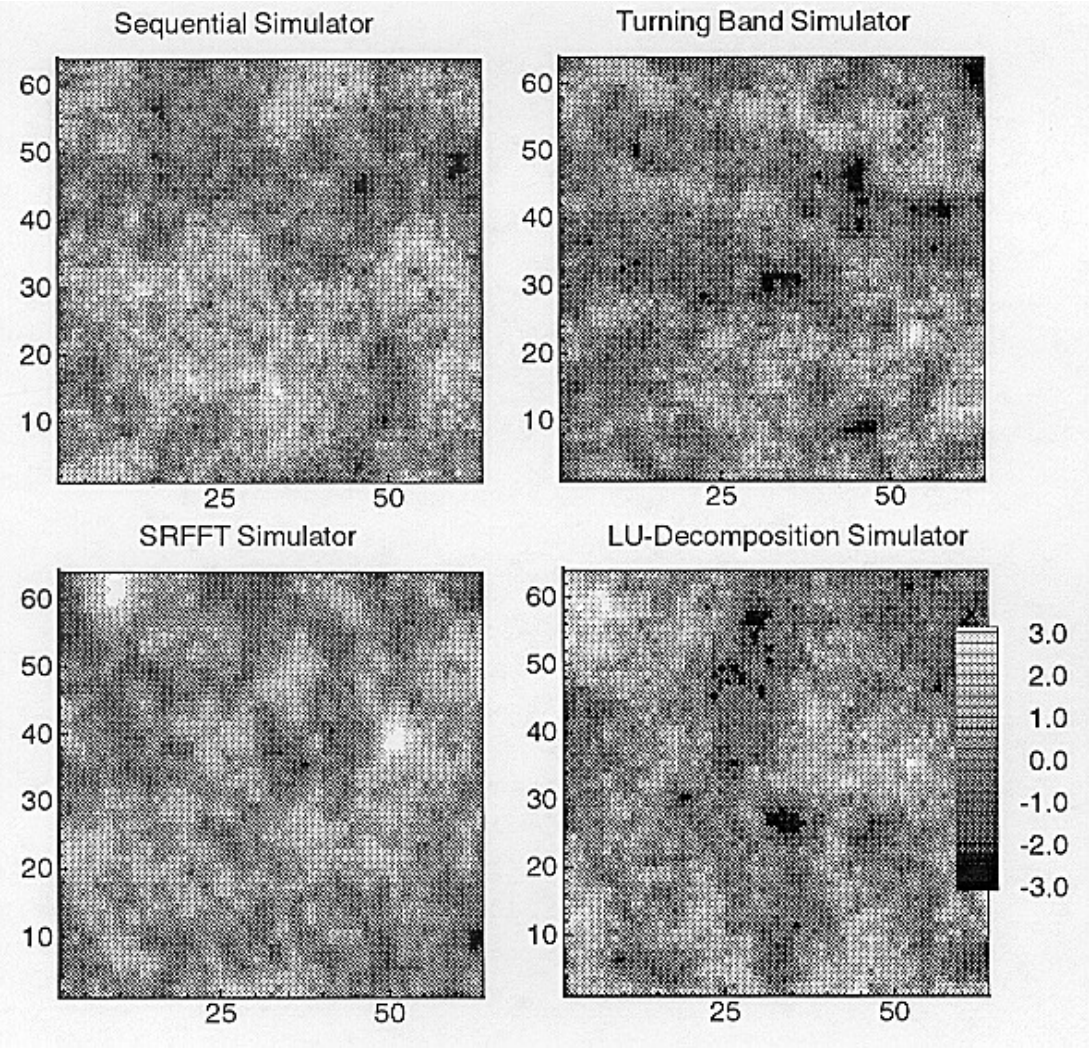

Figure 3.5: Sample Gaussian random field realizations with one realization of each tested RFG. The mean is specified to be 0; the variance is specified to be 1; the correlation function is exponential with $\lambda = 5$.

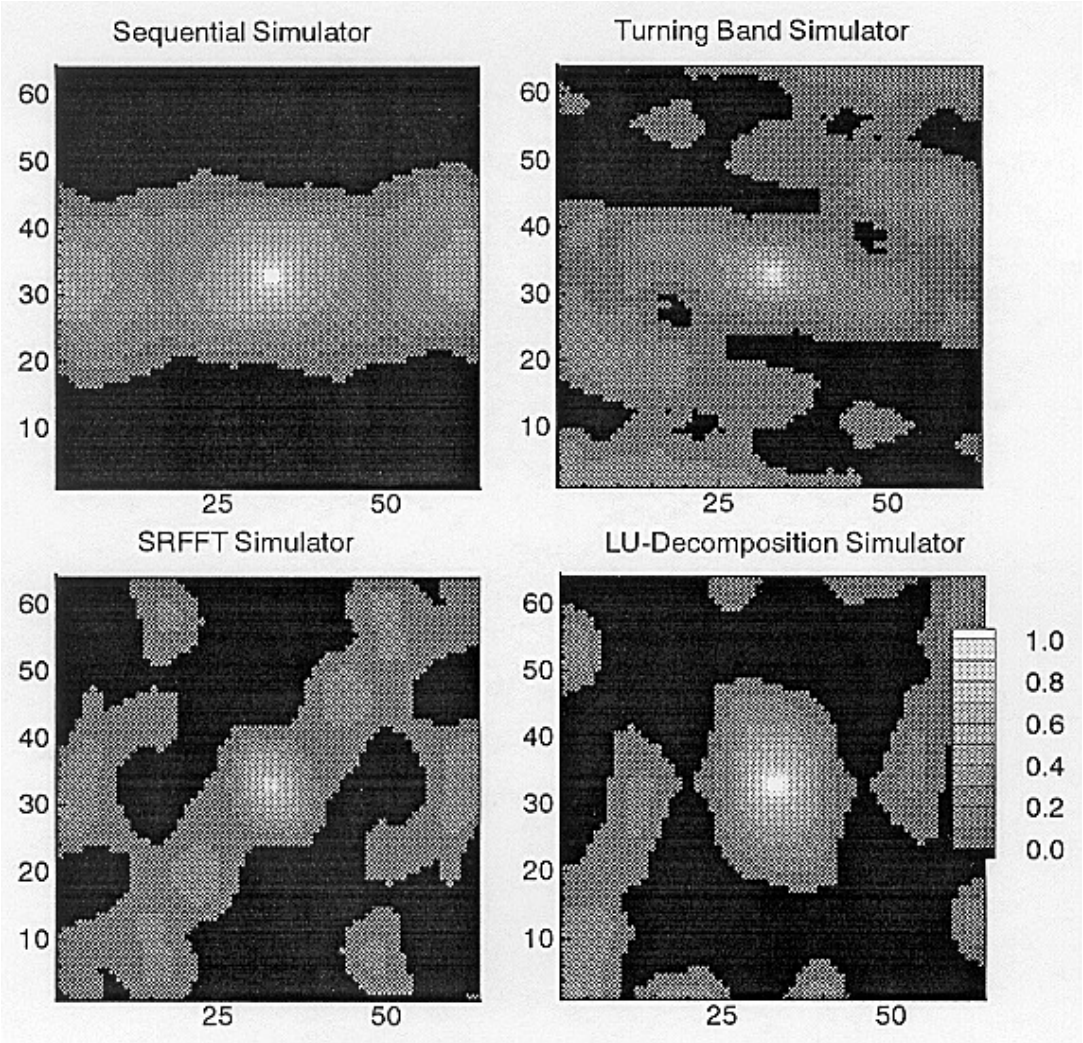

Figure 3.6: Spatial covariance of the sample random field realizations in Figure 3.5 (also see section 3.4.1).

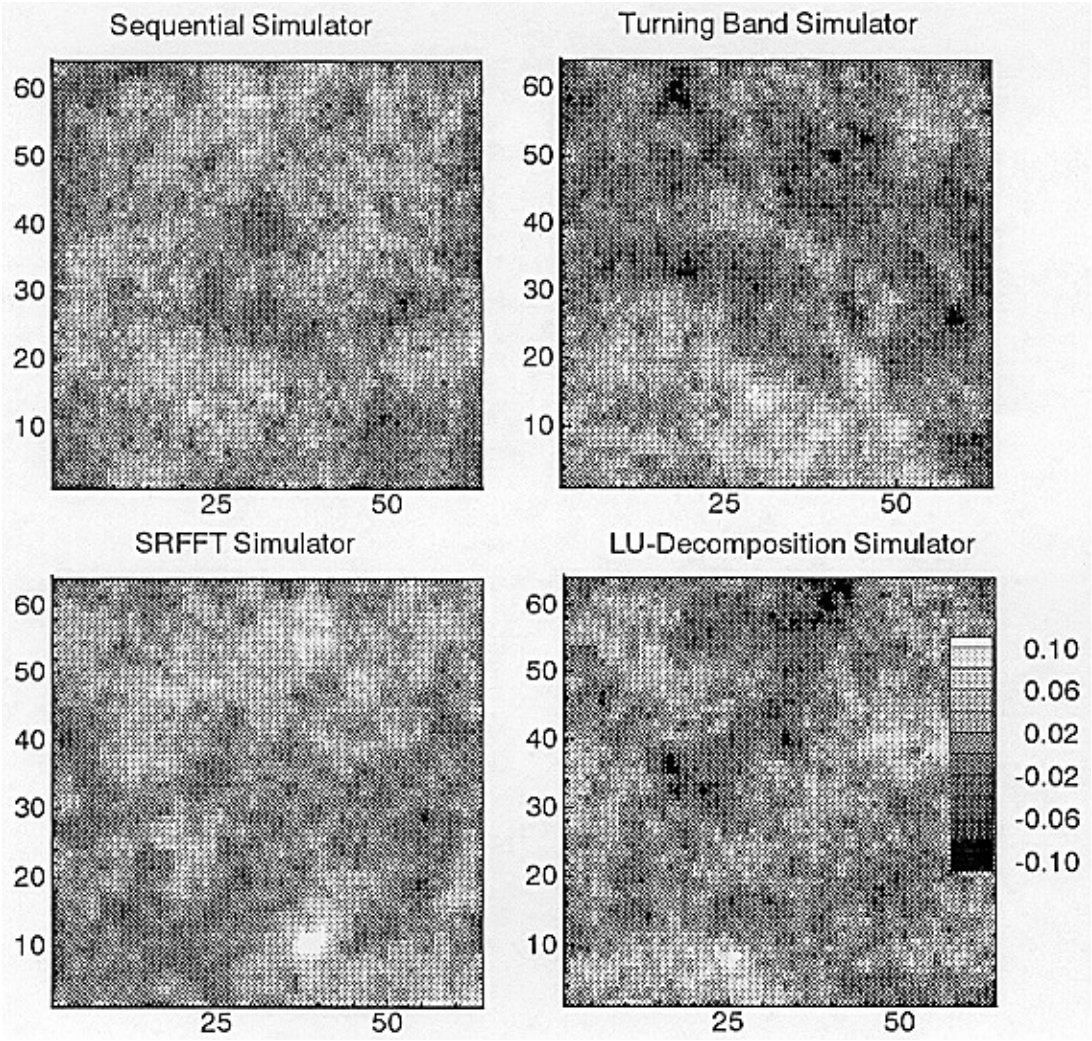

Figure 3.7: Local sample mean of each RFG after 1000 runs.

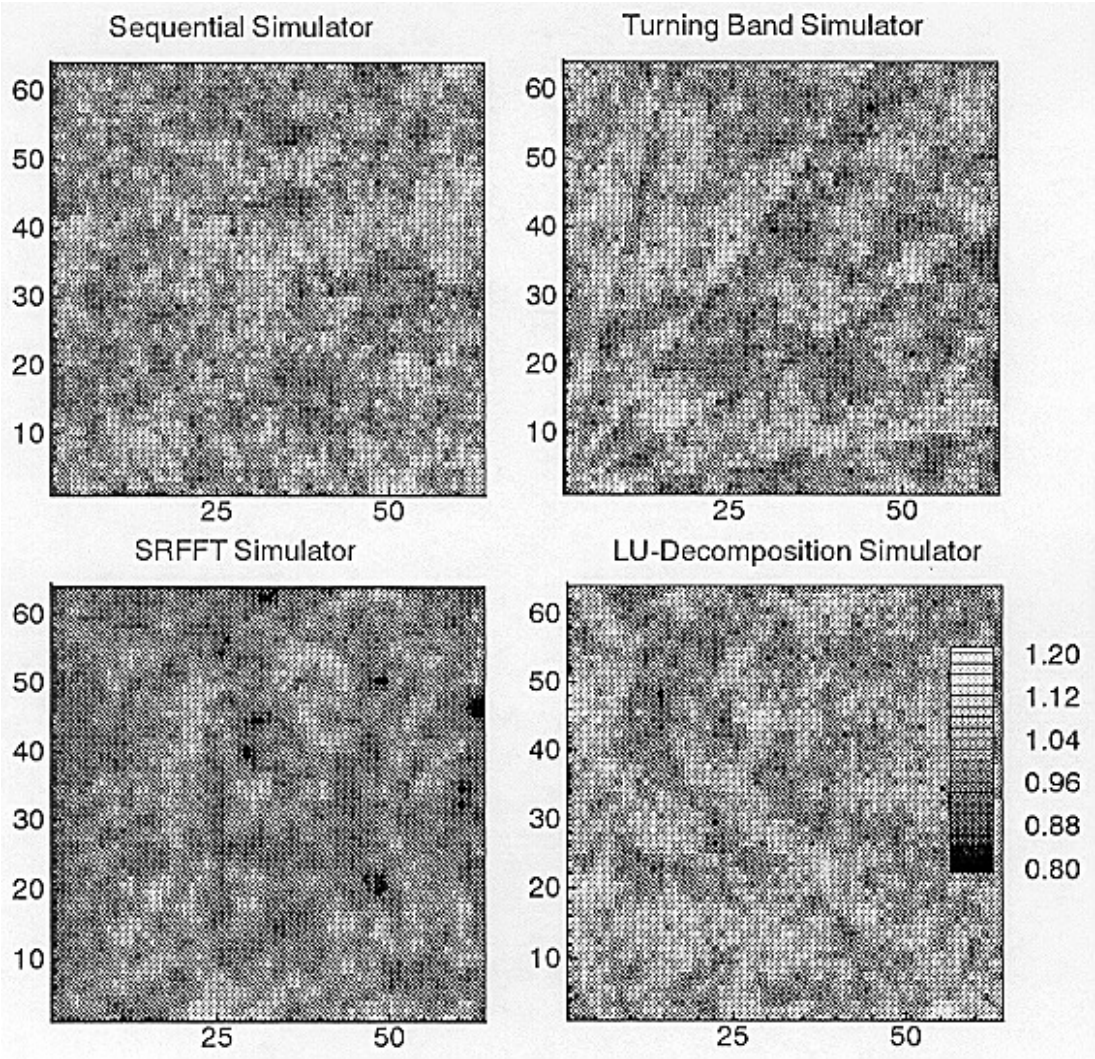

Figure 3.8: Local sample variance of each RFG after 1000 realizations.

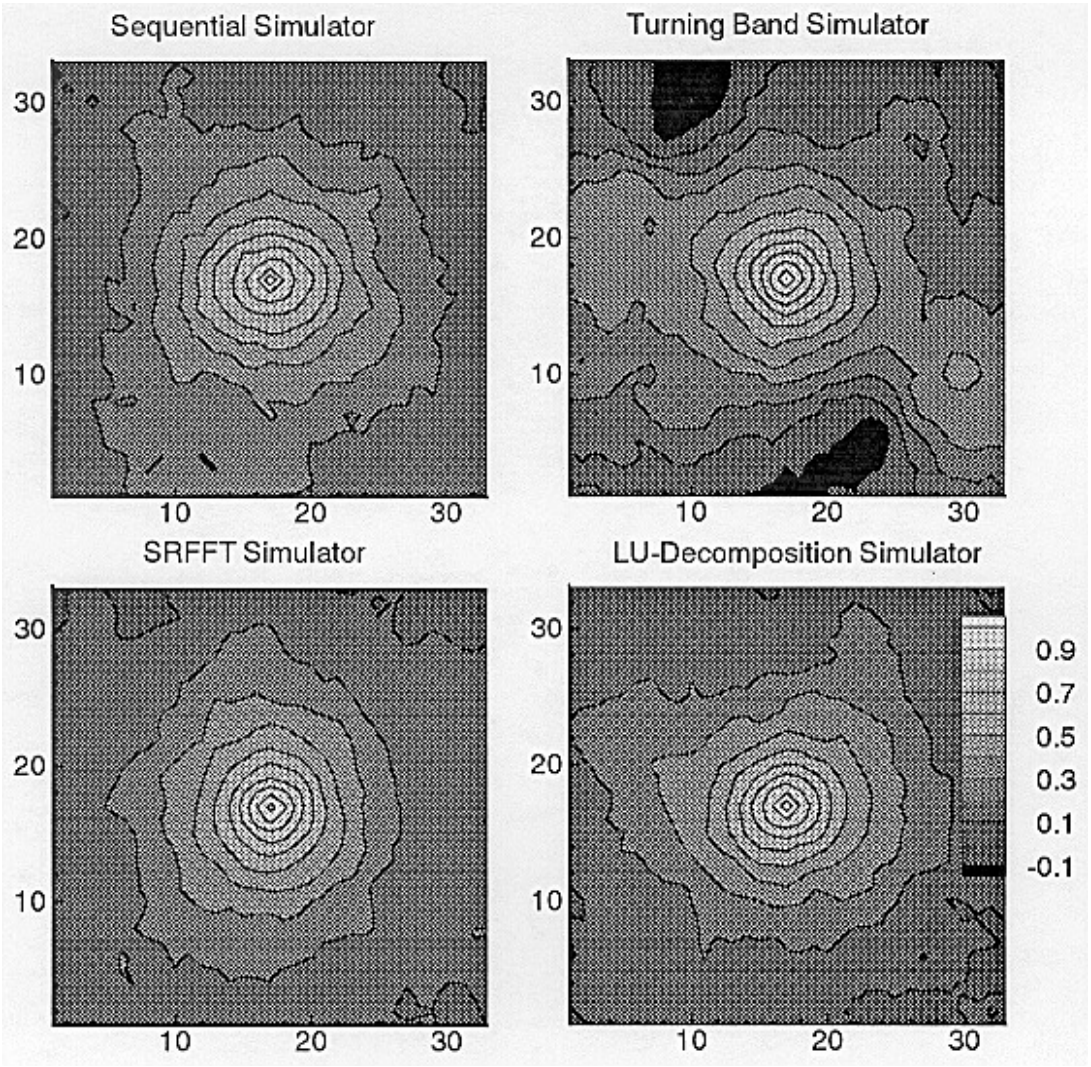

Figure 3.9: Sample local covariance field showing the covariance between point (48,48) and the surrounding 33^2-1 points. The results are based on 1000 realizations.

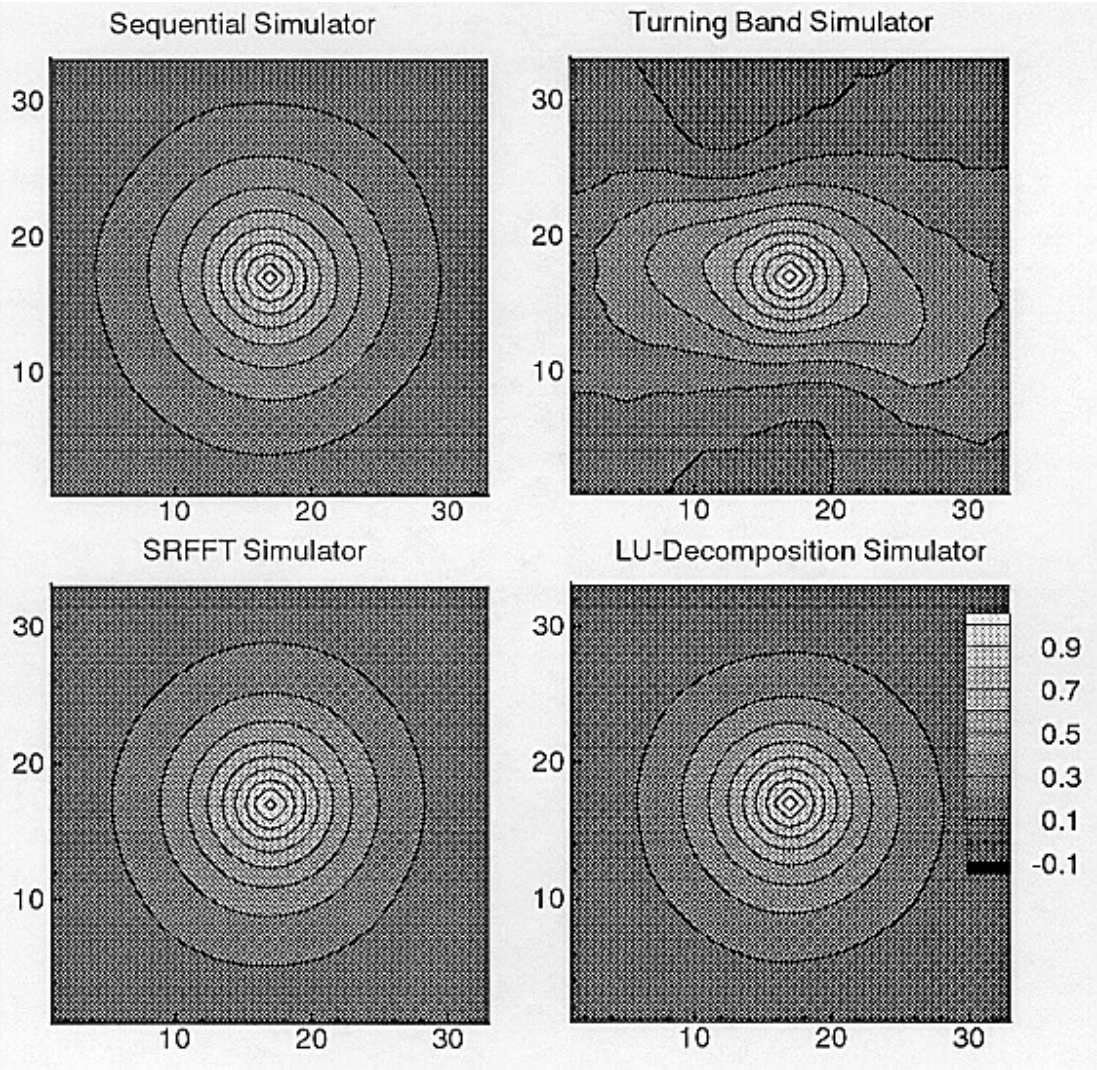

Figure 3.10: Mean of all 312 sample local covariance field (such as those shown in Figure 3.9). Each sample local covariance field is based on 1000 realizations.

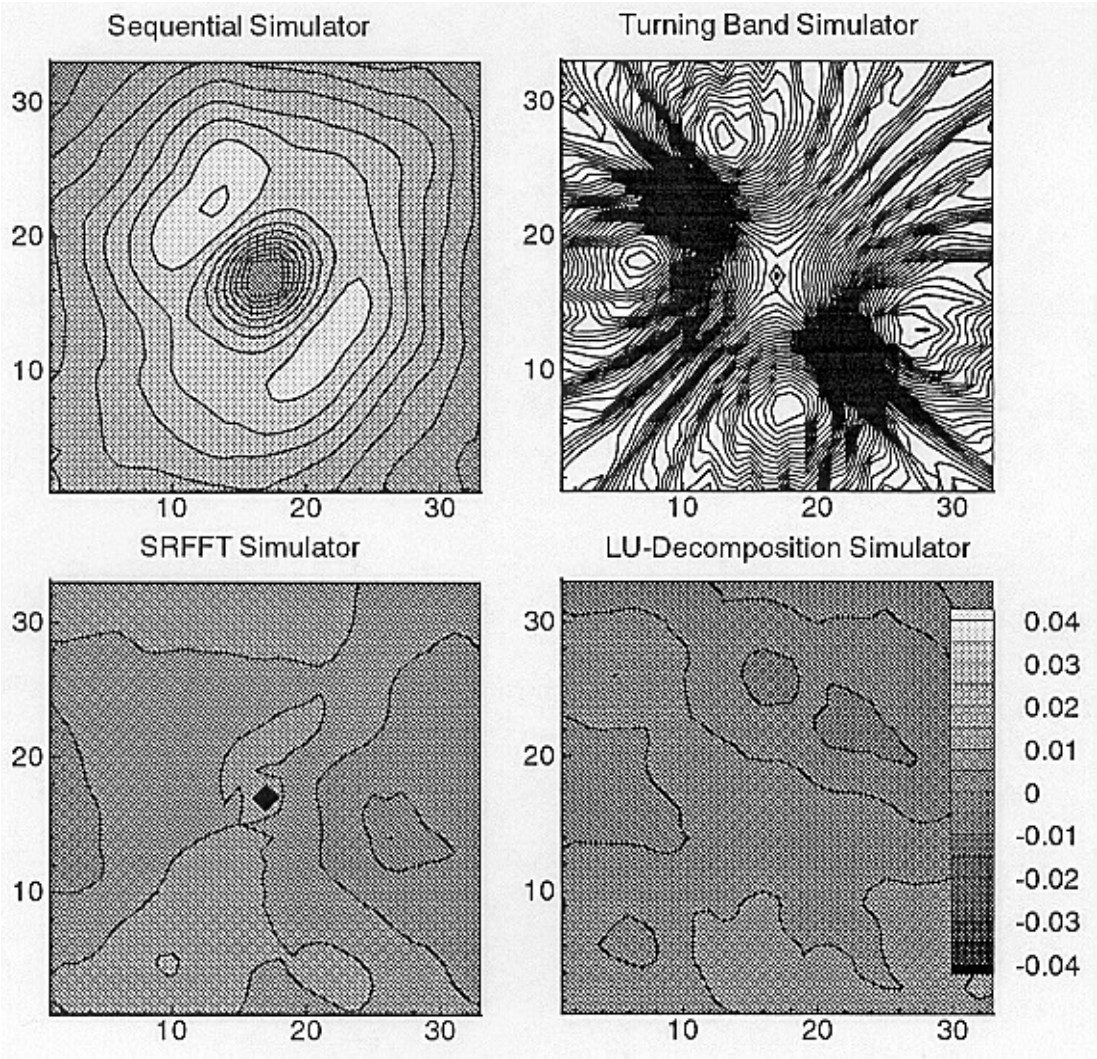

Figure 3.11: Deviation of the mean sample covariance fields (Figure 3.10) from the exponential covariance function. The color flooding is omitted from the plot for the TB simulator. For the TB plot the total number of contour levels has been increased to 61 (instead of 17) to visualized the lineation in the sample covariance. The range of the contours in the TB plot is [-0.13,0.18]. In the TB plot the deviation is negative near the top and bottom of the plot and positive towards the left and right side (see Figure 3.10).

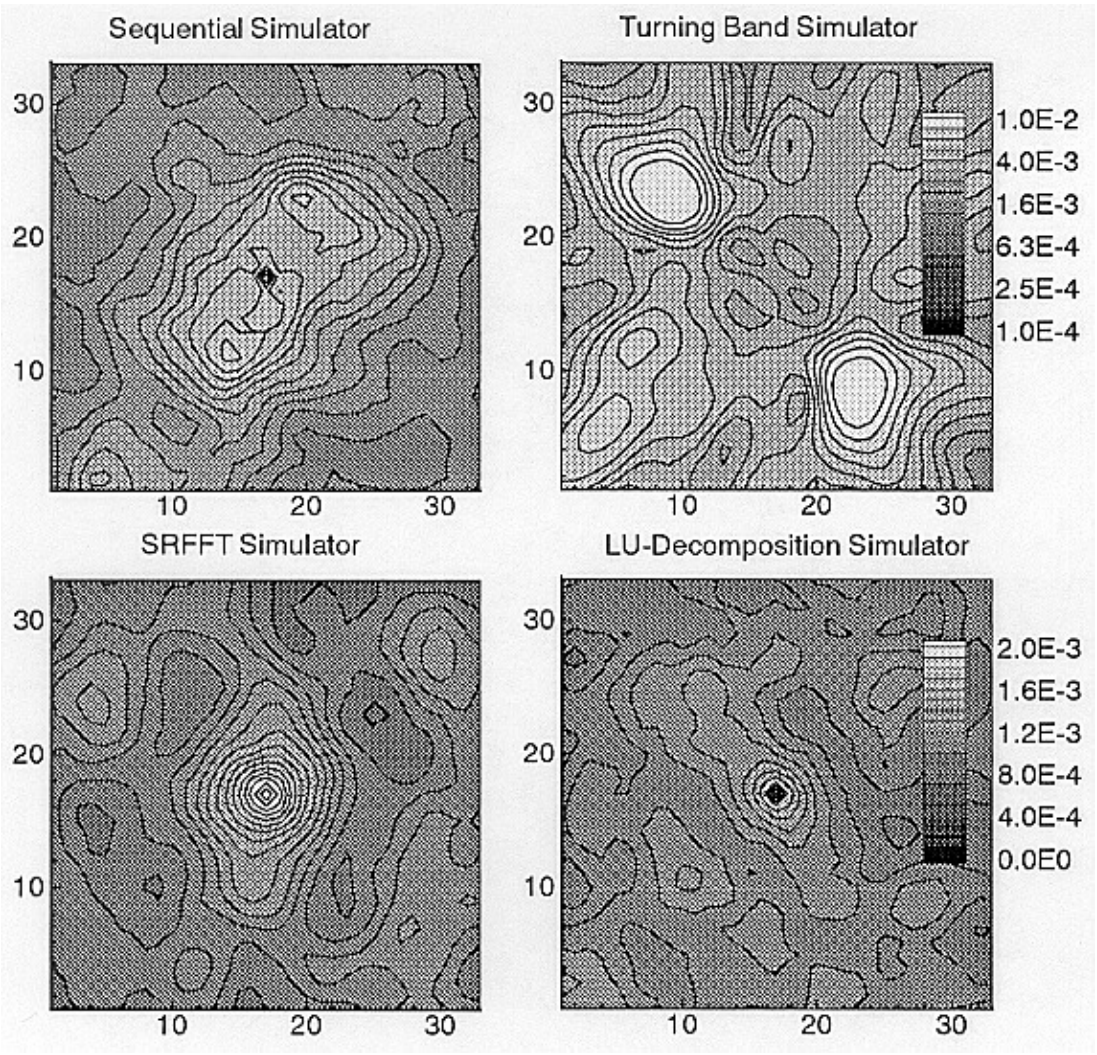

Figure 3.12: Variance of the sample covariance fields. The variance is obtained by superposing all sample covariance fields. Then the variance is computed for each point in the covariance field similar to the local variance of the random fiels. The Turning Band Plot has a different gray-scale than the other three plots (label inserted). It has a larger range and is based on an exponential scale.

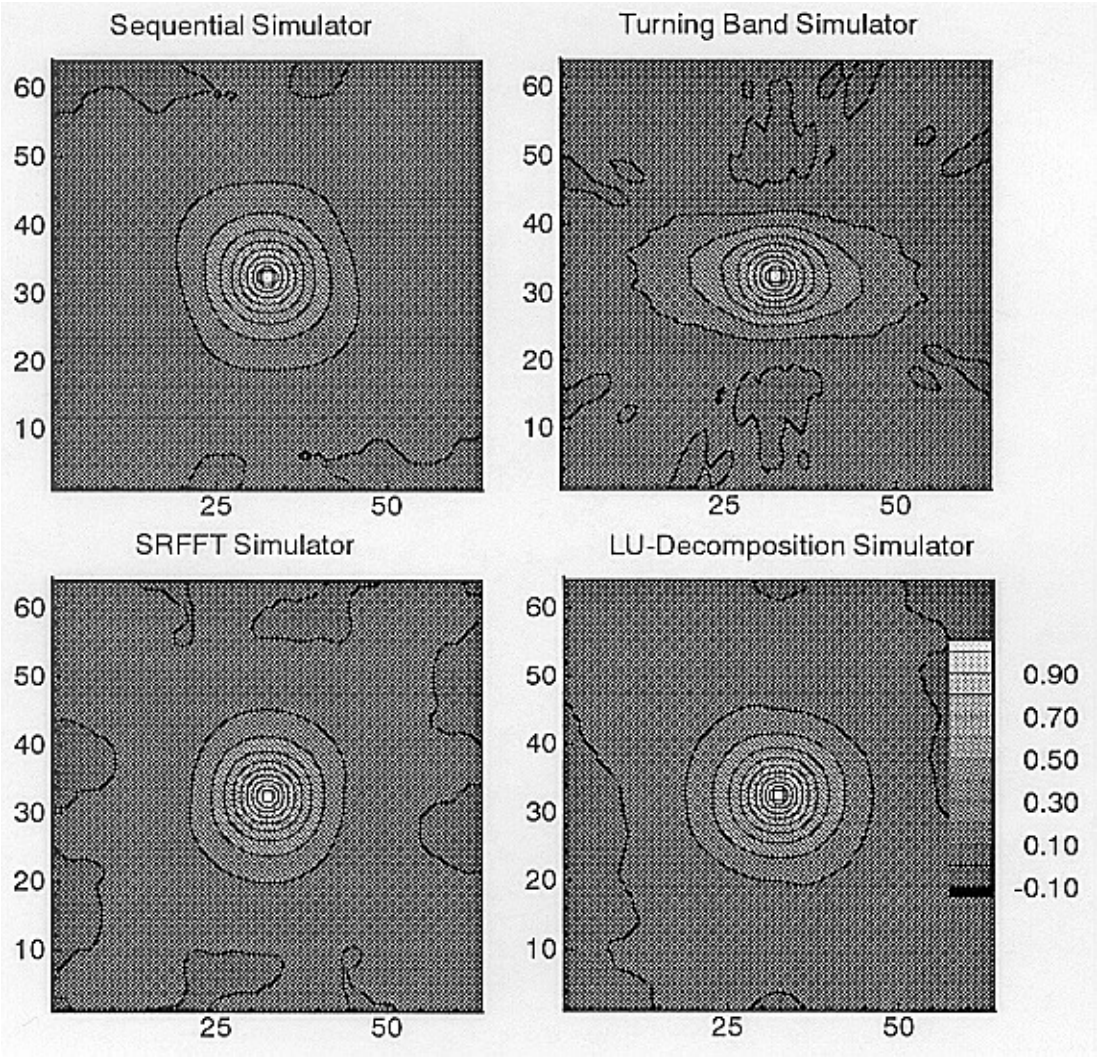

Figure 3.13: Mean of 50 spatial covariance fields such as the ones shown in Figure 3.6.

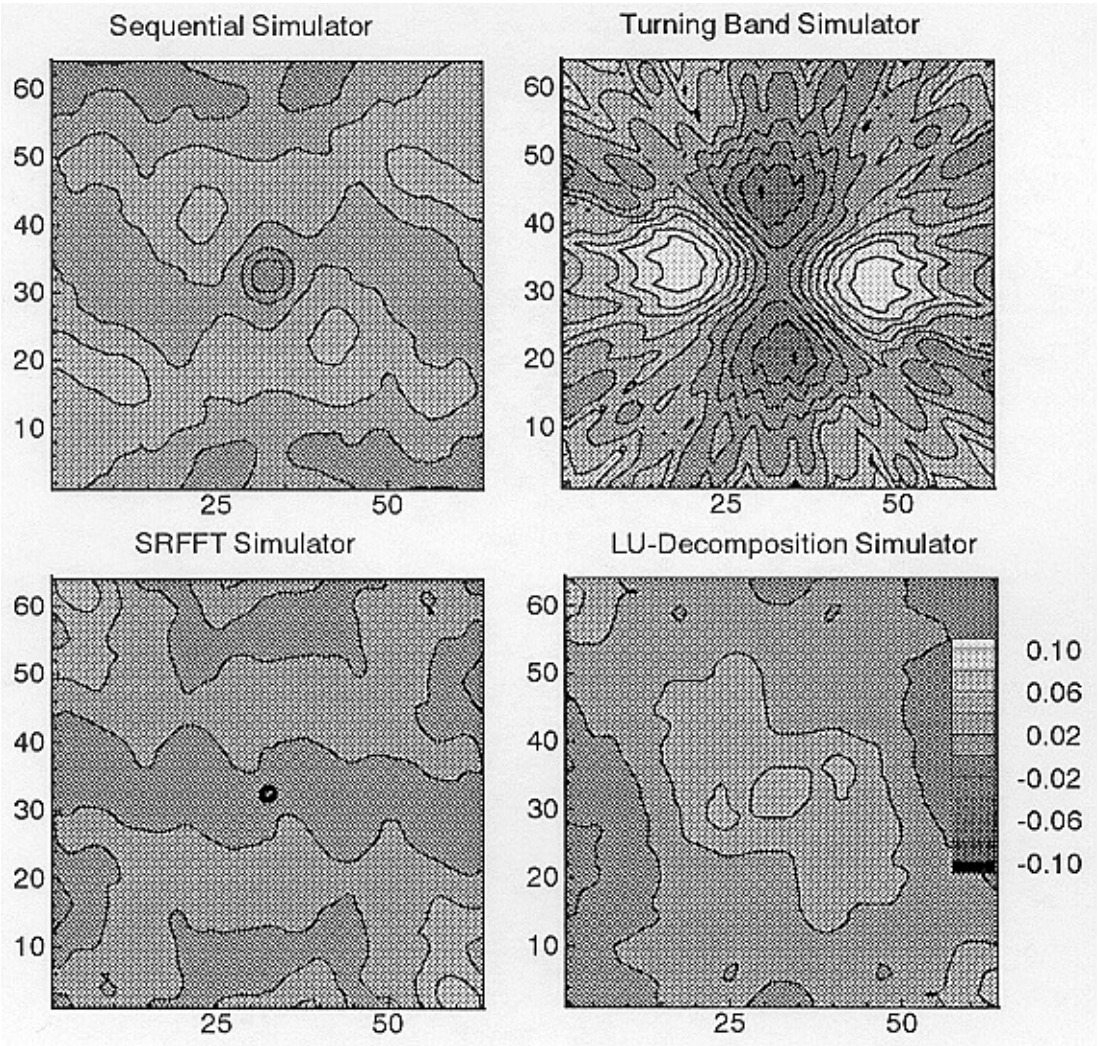

Figure 3.14: Deviation of the mean spatial covariance in Figure 3.13 from the exponential covariance function.

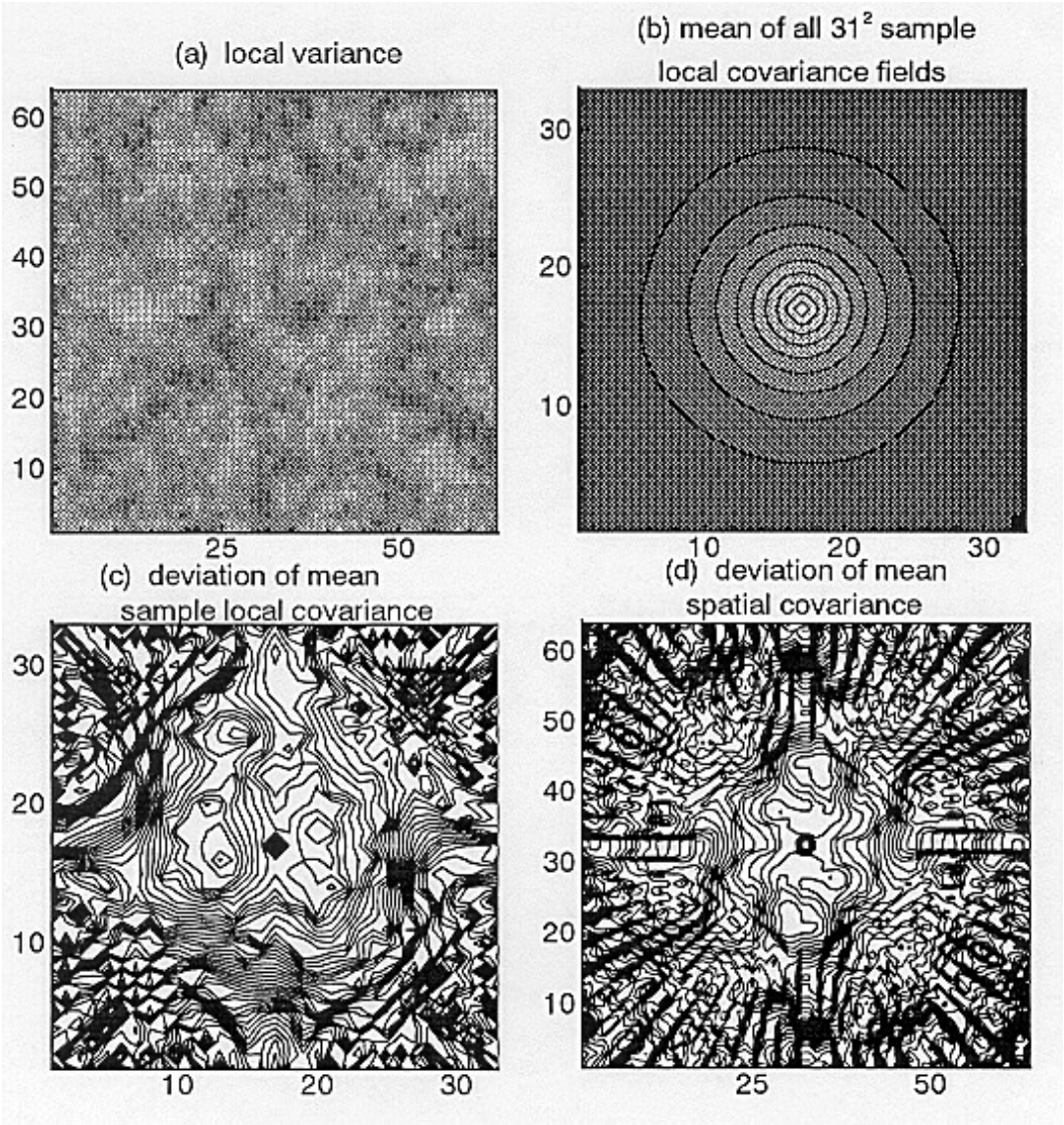

Figure 3.15: Selected results for the improved version of the TB simulator. The plotting variable and the flooding and contour ranges are identical to those of the LU simulator in Figure 3.8(a). Figure 3.10(b), Figure 3.11(c), and Figure 3.14(d). In the latter two plots, the overall range of the deviations is similar to that of the LU simulations. But small lineations remain as shown by the contour lines.