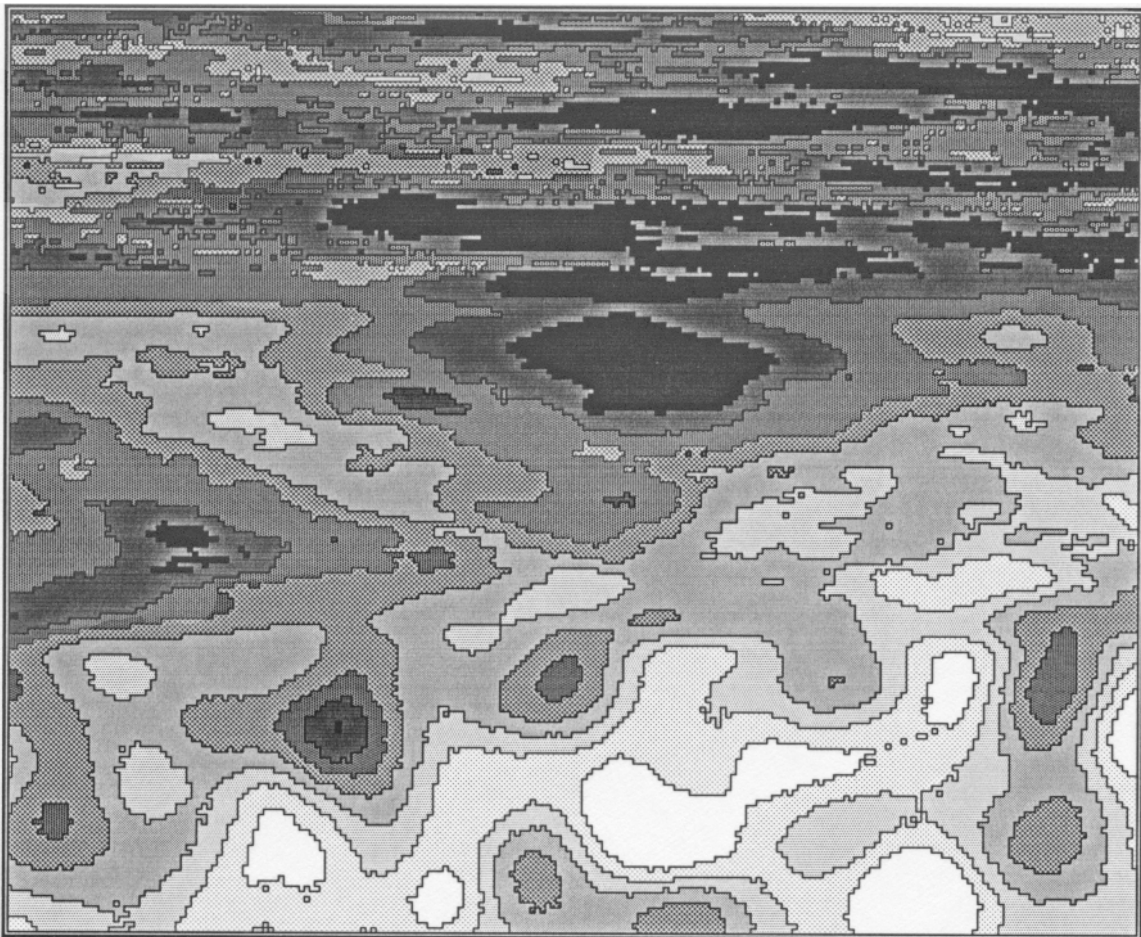


**Description of and User's Manual for TUBA:  
A Computer Code for  
Generating Two-Dimensional Random Fields  
via the Turning Bands Method**

July, 1990



by

D. A. Zimmerman<sup>1</sup> and John L. Wilson<sup>2</sup>

<sup>1</sup>Science and Engineering Technology Applications, Inc., Cedar Crest, New Mexico

<sup>2</sup>New Mexico Institute of Mining and Technology, Socorro, New Mexico

( revised November, 2005 )

### Disclaimer

Neither the developers of this software nor any persons or organizations acting on their behalf make any warranty, express or implied, with respect to this software or assumes any liabilities with respect to its use or misuse, or for the interpretation or misinterpretation of any results obtained from this software, or for any damages resulting from the use of this software.

**Cover:** A synthetic random field representing variations in the hydrologic properties of a layered subsurface environment. The conceptual model of this system and generation of the field is described in section 4.3.4.

Typeset in T<sub>E</sub>X and published by:

***SETA, Inc.***  
**27 Ponderosa Drive, Suite 100**  
**Cedar Crest, New Mexico 87008**  
***<http://www.setainc.com>***

**Description of and User's Manual for TUBA:  
A Computer Code for  
Generating Two-Dimensional Random Fields  
via the Turning Bands Method**

July, 1990

by

D. A. Zimmerman<sup>1</sup> and John L. Wilson<sup>2</sup>

<sup>1</sup>Science and Engineering Technology Applications, Inc. Cedar Crest, New Mexico

<sup>2</sup>New Mexico Institute of Mining and Technology, Socorro, New Mexico

( revised November, 2005 )

Published by

***SETA, Inc.***

27 Ponderosa Drive, Suite 100  
Cedar Crest, New Mexico 87008

<http://www.setainc.com>

## Acknowledgements

A significant amount of credit is owed to Dr. Allan Gutjahr, Department of Mathematics, New Mexico Institute of Mining and Technology, for his support throughout the project, especially with regard to the use of spectral methods for random field generation. We are also grateful to Mr. Warner Losh for developing the tools necessary for producing the plots of the random fields and to Mr. David Becker for his assistance in helping us typeset this document in  $\text{\TeX}$ .

This work was performed under a research grant sponsored by the U.S. Department of Energy; we are grateful for the financial support we received. Our appreciation is also extended to the project officers, Messrs. Jim Chism, E. B. Nuckols, and Edith Allison for their contributions on this project.



## Preface

TUBA, the computer code, was first introduced in 1981. From 1988 to 1990, many new enhancements were added, the code was restructured and this user's manual was written. In the 15 years since the 1990 version of TUBA was released, it has been sent to academic and commercial institutions throughout the world from England to Korea. As this (rebuilding TUBA and writing the user's manual) was a sideline research project during my graduate studies at New Mexico Institute of Mining and Technology, its maintenance eventually fell by the wayside. However, recent interest in the code has provided the motivation to produce a more readily available copy of the user's manual in the currently popular pdf format.

At the time this manual was initially written, the plethora of low-cost graphics software we enjoy today was not available; personal computers were not even in mainstream use at that time! Therefore, the random field plots displayed in the manual had to be generated via low-level, device-dependent C-subroutine calls to a laserjet printer. The original document was produced by running the pages with bitmap graphics through the printer twice – once for the graphic image and the second time for the figure caption and page number! While such graphics software is available today, the time needed to convert all of the output fields to, say, color, shaded-relief plots, can not presently be allocated to this task. Therefore, as an 'interim' measure, all of the original bitmap images and other plots whose source files have been forever lost, have been scanned so they can be incorporated into the 2005 pdf version of the manual. The note at the bottom of the cover page ("revised November 2005") should therefore probably be cast as "revived November 2005."

There remain minor inconsistencies between some of the input files shown in the manual, originally written for TUBA version 2.0 released in 1990, and the input required by the latest version of the code, TUBA version 2.11. However, all of the inputs to the current version are described in the addendums to Chapter 4, under TUBA version 2.10 Documentation and TUBA version 2.11 Documentation. It is important to read these sections as they describe several user-requested code enhancements.

There have been plans to develop a GUI for TUBA, complete with random field plotting, conditional simulation capabilities and HTML Help documentation. These plans also include rewriting the manual, updating the printed input files to reflect the current version of the code, incorporating version 2.10 and 2.11 documentation into the appropriate places in Chapter 4, and producing the graphics in color. However, whether and when these plans will actually materialize is undetermined at this time. Until then, I hope you find the current version to be a useful tool.

D. A. (Tony) Zimmerman, P.E.  
SETA, Inc.



# Table Of Contents

## **Chapter 1: Introduction**

1.1 Code Purpose .....	1
1.2 Summary of Code Capabilities .....	2
1.3 Code History .....	2
1.4 Scope of Report .....	3

## **Chapter 2: Overview of Theory**

2.1 Statistical Characterization of Random Fields .....	5
2.1.1 Fundamental Concepts .....	5
2.1.2 Estimation of Random Field Statistics .....	10
2.2 The Turning Bands Method .....	13
2.3 Spectral Line Generation Methods .....	16
2.4 The Moving Average Process and the Telis Covariance .....	21
2.5 Generalized Covariances and Intrinsic Random Functions .....	24
2.6 Simulation of Areal Average Random Fields .....	26
2.7 Simulation of Anisotropic Random Fields .....	28

## **Chapter 3: Line Process Generation – Practical Aspects**

3.1 Geometrical Considerations .....	29
3.2 Spectral Generation of the Line Process .....	30
3.2.1 Tests to Determine the Effect of Spectral Approximations .....	30
3.2.2 Effect of Truncating the Spectrum .....	31
3.2.3 Effect of Discretizing the Spectrum .....	32
3.3 Moving Average Generation of the Line Process .....	34

## **Chapter 4: User's Manual**

4.1 Code Input Description .....	35
4.1.1 Output Field Parameters .....	35
4.1.2 Covariance Model Parameters .....	36
4.1.3 Turning Band Parameters .....	37
4.1.4 Output File Parameters .....	39
4.1.5 Simulation Parameters .....	39

## **User's Manual continued**

4.2	Example Runs – Input, Output, and Analysis .....	40
4.2.1	Generation of Stationary Isotropic Random Fields.....	42
4.2.2	Generation of Anisotropic Random Fields.....	51
4.2.3	Generation of Non-Stationary Random Fields.....	53
4.2.4	Generation of Areal Average Processes.....	57
4.2.5	Generation at Arbitrary Points in Space .....	61
4.2.6	Generating Subregions at Higher Resolution.....	64
4.3	Some Practical Aspects Illustrated.....	69
4.3.1	Sample Statistics versus Target Statistics .....	69
4.3.2	Representing Spatial Variability Patterns .....	70
	The Mask File Option.....	74
4.3.3	Using the Spectral Method of Shinozuka and Jan.....	77
4.3.4	Hydrology Application – a layered System .....	80
4.3.5	Designing Christmas Cards with TUBA.....	82
4.3.6	An Important Note About Generating Large Fields.....	85
	<b>TUBA version 2.10 Documentation .....</b>	<b>89</b>
	Code Modifications.....	90
	Reproducing the “ith” Simulation Using Two RNG Seeds.....	92
	Generating Onto An Irregularly-spaced Finite-Difference Grid .....	94
	The Ergodicity Of The Turning Bands Algorithm .....	96
	<b>TUBA version 2.11 Documentation .....</b>	<b>97</b>
	Memory Requirements Versus Execution Speed.....	98
	Specifying Generalized Covariance Model Parameters.....	98
	Specifying Turning Bands Parameters.....	99
	New Simulation Parameters Options.....	99
	Effect of Change in Default Turning Bands Line Discretization Length.....	100
	<b><u>Chapter 5: Programmer's Manual</u></b>	
5.1	Redimensioning.....	101
5.2	Code Structure and Programming Notes .....	102
5.3	Random Number Generators and the Fast Fourier Transform.....	104
5.4	User-defined Covariance Models .....	105
	<b>References .....</b>	<b>107</b>
	<b>A reprint of <i>Mantoglou and Wilson</i>, [1982] .....</b>	<b>Appendix A</b>
	<b>TUBA version 2.11d Computer Code Listing .....</b>	<b>Appendix B</b>

## List of Figures

Figure 1.	Two-dimensional covariance models . . . . .	7
Figure 2.	Schematic of the Turning Bands algorithm . . . . .	15
Figure 3.	Radial spectral density functions . . . . .	17
Figure 4.	Spectral distribution functions . . . . .	18
Figure 5.	Schematic of the Moving Average process . . . . .	22
Figure 6.	Effect of frequency spacing on correlation structure of line process . . . . .	33
Figure 7.	Effect of discretization interval on correlation structure of line process . . . . .	34
Figure 8.	Plot of Gaussian random field . . . . .	46
Figure 9.	Plot of Bessel random field . . . . .	47
Figure 10.	Plot of exponential random field . . . . .	48
Figure 11.	Plot of Telis random field . . . . .	49
Figure 12.	Variogram estimates of TUBA generated fields . . . . .	50
Figure 13.	Variogram estimates for anisotropic Gaussian field . . . . .	51
Figure 14.	Plot of anisotropic Gaussian random field . . . . .	52
Figure 15.	Plot of an Intrinsic random field of order-0 . . . . .	54
Figure 16.	Plot of an Intrinsic random field of order-1 . . . . .	55
Figure 17.	Plot of an Intrinsic random field of order-2 . . . . .	56
Figure 18.	Exponential field with areal averaging = $\frac{1}{4}th$ correlation length . . . . .	58
Figure 19.	Exponential field with areal averaging = the correlation length . . . . .	59
Figure 20.	Exponential field with non-overlapping areal averaging . . . . .	60
Figure 21.	Finite element grid superimposed on finite difference grid . . . . .	63
Figure 22.	Alignment of block- and point-centered grids . . . . .	67
Figure 23.	Gaussian field with high-resolution subregion . . . . .	68
Figure 24.	A random field at three different levels of resolution . . . . .	71
Figure 25.	Gaussian field plotted as contours and shaded terraces . . . . .	72
Figure 26.	Exponential field plotted as contours and shaded terraces . . . . .	73
Figure 27.	Exponential field plotted using profiles and terraced contours . . . . .	75

Figure 28. Plan and perspective views of an exponentiated Gaussian field.....	76
Figure 29. Gaussian field demonstrating the mask file option .....	78
Figure 30. Exponential field generated via the method of Shinozuka and Jan.....	79
Figure 31. Plot of a synthetic layered hydrogeologic system.....	81
Figure 32. Schematic of where not to place the Turning Bands origin .....	83
Figure 33. Random field generated by a mad scientist.....	84
Figure 34. Exponential field spanning 200 correlation lengths.....	86
Figure 35. Exponential field generated using only four Turning Band lines .....	87
Figure 36. Point-Centered, Irregularly-Spaced Finite Difference Grid.....	95
Figure 37. Order of subroutine calls in TUBA .....	103
Figure 38. Graphical “test” of random number generator.....	106

## List of Tables

Table 1. 2D covariance models and their radial spectral density functions .....	10
Table 2. Polynomial Generalized Covariance models in TUBA .....	26
Table 3. Effect of truncating the spectrum on the variance of the fields .....	32
Table 4. Spectral parameters for generating the line process.....	33

# Chapter 1

## Introduction

TUBA is a computer code for generating synthetic two-dimensional random fields via the Turning Bands Method. TUBA was first introduced by *Mantoglou and Wilson*, [1981]. Some aspects of the code are described in *Mantoglou and Wilson* [1982] and the code is reviewed in the textbook by *Bras and Rodriguez* [1984]. Since 1981, numerous requests for TUBA have been made, and it was in response to those requests that this report was compiled. The overall goal in compiling this report was, therefore, to produce a fully documented computer code that is directed toward the *application* of this technology. The primary objectives of this work were to (1) provide a “refurbished” version of the computer code, (2) incorporate additional options which expand its usefulness and improve its efficiency, (3) review the theory behind the algorithms, (4) discuss important practical aspects that facilitate the proper application of the code, and (5) provide documentation with a user’s manual and a programmer’s manual. The 1990 version of the code was called version 2.0; the current version is version 2.11.

### § 1.1 Code Purpose

The synthetic generation or simulation of two-dimensional random fields has many applications in the geophysical sciences of mining, petroleum engineering, and hydrology [see e.g., *Warren and Price*, 1961; *Mejia and Rodriguez-Iturbe*, 1974; *Journel and Huijbregts*, 1978; *Smith and Freeze*, 1979]. The main application areas in hydrology are the synthetic generation of rainfall over an area or the spatial distribution of runoff or aquifer properties. In petroleum engineering, the main application is the generation of reservoir properties, while in mining it is ore grades. The ability to generate these random fields is important because it is very difficult to both measure and characterize the in situ distribution of these highly variable quantities (e.g., ore grade, permeability, porosity). In hydrology and petroleum engineering, the ability to simulate synthetic random fields provides a means of assessing the influence (and uncertainty) of the spatial variability in the permeability on the head or pressure of a flow field, and the solute concentration of miscible displacement, through stochastic models. This type of analysis has recently been carried out via Monte Carlo simulations of the physical processes [e.g., *Smith and Freeze*, 1979; *Delhomme*, 1979; *Smith and Schwartz*, 1981 and many others]. Here, multiple realizations of a random input process having the statistical properties of the

true (hypothesized or estimated) underlying field are used sequentially in a deterministic hydrologic or petroleum simulation model to obtain a series of output processes whose statistics can be calculated and related to the input process. Examples of the application of the Turning Bands method for simulation of natural processes can be found in *Journal* [1974], *Journal and Huijbregts* [1978], *Delhomme* [1979], and *Tompson et al.* [1989]. Examples of TUBA 1.0 applications are found in *Frind et al.* [1987], *Molissis* [1988], *Sudicky and MacQuarrie* [1989], *MacQuarrie and Sudicky* [1989], *Sudicky et al.* [1989], *Kueper et al.* [1989 and 1990], *Sampier and Neuman* [1989], *Wagner and Gorelick* [1989], *Gomez-Hernandez and Gorelick* [1989], and *Rubin and Gomez-Hernandez* [1990].

### § 1.2 Summary Of Code Capabilities

TUBA is a computer code for generating synthetic two-dimensional stationary or non-stationary random fields. Random fields are commonly characterized by their statistical properties (e.g., mean, variance and covariance). Five functional forms for the covariance structure are available in TUBA, these are: exponential, Gaussian, Bessel, Telis, and Generalized Covariance models. Other forms can be supplied by the user. TUBA can generate isotropic or anisotropic random fields and can simulate areal average processes. The random fields can be generated onto a gridded system (e.g., at the grid points or nodes of a block *or* point-centered finite difference grid) or at arbitrary locations in space (e.g., at the gauss points of a finite element grid). TUBA can be used to generate the field values in local areas at much greater resolution than the original simulated field. The fields can be generated with a normal or a lognormal distribution. The size of the simulation is limited only by the virtual memory capabilities of the computer it is run on; random fields with over one million nodes have been generated with TUBA using a PC.

### § 1.3 Code History

TUBA was originally written in Fortran IV by Aristotelis Mantoglou while he was a Masters student at MIT in 1981; with this new version, the code has been modularized, vectorized, converted to Fortran 77, expanded and modified to run more efficiently. Modularization (breaking the code into subroutines) enhances the readability of the code and enables persons familiar with the algorithms to understand (and perhaps modify) the program. Vectorization means that input data arrays, internal working arrays, and output field data arrays are all addressed through a single vector. This permits greater flexibility in choosing the type and size of problem to be run while reducing computational effort by minimizing paging (swapping pages of memory to and from



disk). It also provides a convenient way to redimension the code by changing a single parameter statement. Conversion to Fortran 77 really means conversion to a particular programming style while taking advantage of recent Fortran enhancements which are not available on Fortran IV compilers. Input and output routines have been improved with respect to ease of use and flexibility and new options were added; these include: the ability to generate field values at arbitrary locations in space (as opposed to only gridded output) and a more general ability to generate anisotropic random fields.

TUBA was also modified to run more efficiently by incorporating a Fast Fourier Transform (FFT) algorithm for spectral generation of the line processes. Other line process algorithms in the code include a spectral method, a moving average method, and a Brownian motion method. Finally, an FFT subroutine and two random number generator subroutines were added so that TUBA requires no external libraries or compiler dependent intrinsic subroutines in order to run.

## § 1.4 Scope of Report

This report provides a condensed review of the theory behind the algorithms used in TUBA, a discussion of important practical considerations, a description of how to run the program, and some documentation of the Fortran code itself. Chapter 2 reviews some fundamental concepts associated with the statistical characterization of random fields and reviews the theory behind the algorithms used in TUBA. Chapter 3 provides important guidelines of practical importance related to the line process generation and its effect on the accuracy and efficiency of the simulations. Chapter 4 (the User's Manual) provides a description of the code input and provides general guidelines on how to properly select a consistent set of input values. It also contains examples of how to choose input values for specific cases, illustrating the interactive or batch session, and shows the results (plots, statistical analyses) of the simulations. Chapter 5 (the Programmer's Manual) discusses aspects associated with implementation of TUBA on different computer systems, describes the structure of the Fortran code, and explains some operational features of certain parts of the code. Appendix A contains a reprint of *Mantoglou and Wilson*, [1982]. A listing of the TUBA computer code is provided in Appendix B. The source code and executable can be downloaded from <http://www.setainc.com/tuba>.



## Chapter 2

### Overview Of Theory

This chapter begins with a review of some fundamental concepts associated with the statistical characterization of random fields. This leads to an understanding of the motivation for the type of synthetic random fields generated by TUBA. Following this, a cursory review of the theory behind the Turning Bands method and algorithms ancillary to the Turning Band method is presented. Finally, a discussion of intrinsic random fields, areal average processes, and anisotropic random fields is presented.

#### § 2.1 Statistical Characterization of Random Fields

This section begins with an explanation of some basic concepts that are essential for understanding how the complex behavior of random fields can be described mathematically. The section ends by describing some techniques for estimating the statistics of discrete random fields.

##### § 2.1.1 Fundamental Concepts

A random field can be conceptualized as a collection of random variables ordered in space – a spatial stochastic process. Each random variable has its own probability density function, thus a complete description of the process requires knowledge of the joint probability distribution over the entire domain. Practically speaking, this is impossible to obtain, particularly when only one realization of the field is available. Therefore, the process is usually described by a limited number of statistical parameters (called moments) which characterize the probability density functions.

##### Gaussian Processes

The probability distribution of a random variable or process can be described exactly if all of its moments are known. The statistical moments are defined by

$$E[Z^n] = \int_{-\infty}^{+\infty} z^n f(z) dz \quad n = 0, 1, 2, \dots$$

where  $E[ ]$  denotes expectation,  $Z$  is the random variable, and  $f(z)$  is its probability density function. A Gaussian process (i.e., one that is normally distributed) can be completely described from knowledge of its first two moments (the mean,  $m$ , and the

variance,  $\sigma^2$ ). This is because all higher order odd moments are zero and all higher order even moments depend only on  $\sigma^2$  [Vanmarke, 1984]. Fortunately, a great many natural processes can be described as having Gaussian behavior [Neuman, 1982]. However, even with a Gaussian process, we are still faced with the task of determining  $m$  and  $\sigma^2$  for each point in the field – again, an impractical task when based on field measurements and an impossible task when there is only one realization. There are, however, a few other key properties of random fields which make this problem tractable.

### Covariances

The very concept of a “random field” is nothing more than a mathematical abstraction which is used to describe a complex natural process, such as a heterogeneous permeability distribution in an aquifer or petroleum reservoir. Early efforts to obtain statistical descriptions of aquifer heterogenities focused on the frequency distribution approach where the probability behavior of the material variations is determined from a histogram of the measured values. An inherent assumption associated with this approach is that material property variations are governed by the same probability law at every point in space; i.e., the measurements are assumed to be independent of each other and hence lack correlation, even if they are located in close proximity. It is quite obvious, however, from observable features in natural deposits, that material properties at adjacent points *will* be related. The covariance function,  $C(\vec{x}_1, \vec{x}_2)$ , is used to characterize this spatial dependence between two points in the field, and is defined as

$$C(\vec{x}_1, \vec{x}_2) \equiv E[Z(\vec{x}_1)Z(\vec{x}_2)] = \int_{-\infty}^{+\infty} Z(\vec{x}_1)Z(\vec{x}_2)f(\vec{x}_1, \vec{x}_2) d\vec{x} \quad (1)$$

where  $\vec{x} = (x, y)$  is a location vector,  $Z(\vec{x})$  is the random field with mean zero and  $f(\vec{x}_1, \vec{x}_2)$  is the joint probability density function of  $Z(\vec{x})$  at the points  $\vec{x}_1$  and  $\vec{x}_2$ . As the lag or separation distance between the two points becomes large, they become less correlated ( $f(\vec{x}_1, \vec{x}_2) \rightarrow 0$ ), hence their covariance tends to zero. The covariance function takes on its maximum value at zero lag where it is equal to the variance of the process. A plot showing four different covariance functions, all of which are programmed into TUBA, is shown in Figure 1. This spatial covariance structure is sometimes directionally dependent in which case it is said to be an anisotropic covariance. For example, in a sedimentary deposit the permeability field is likely to be correlated over much greater distances in the horizontal plane than the vertical direction. Frequently, the covariance behavior is assumed to be the same at every point in the field, which leads to the concept of statistical homogeneity.

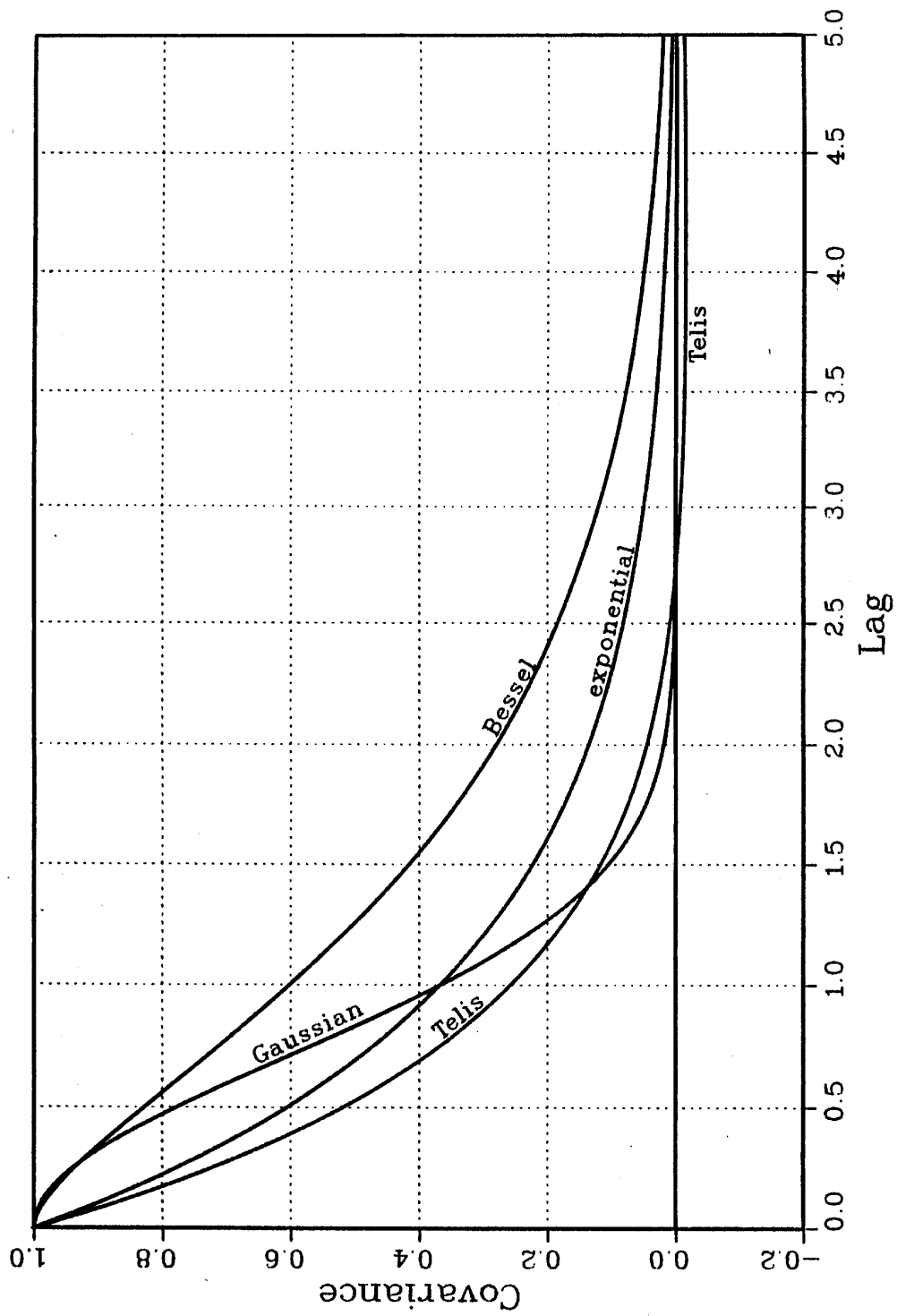


Figure 1. Two-dimensional covariance models programmed into TUBA. The equations for these models are shown in Table 1.

### Stationarity/Statistical Homogeneity

The property of being statistically homogeneous or stationary “in the strict sense” means that the joint probability behavior among a set of variables remains invariant under translation (but not rotation) i.e., the probabilities depend only on their relative positions, not their absolute location. Since, in practice, only the first two moments are estimated, the process is assumed to possess this property “in the weak sense” and is said to be weakly stationary or second-order stationary. Most often the term stationary is used with weak stationarity being implied. A random process  $Z(\vec{x})$  is said to be second-order stationary if

$$\begin{aligned} E[Z(\vec{x})] &= m \\ E[Z'(\vec{x})Z'(\vec{x} + \vec{\xi})] &= C(\vec{x}, \vec{x} + \vec{\xi}) = C(\vec{\xi}) \end{aligned}$$

where  $m$  is a constant,  $Z'(\vec{x}) \equiv Z(\vec{x}) - m$ , and  $\vec{\xi}$  is the vector separation between the points  $\vec{x}$  and  $\vec{x} + \vec{\xi}$ . In other words, if the mean is constant and the covariance between two points depends only on their vector separation and not their absolute locations, then the process is (second-order) stationary. Thus, for stationary Gaussian processes, we can characterize the joint probability behavior from estimates of the mean and covariance function based on measurements taken at various points in space. If the mean varies in space, we can subtract it from the process to leave a mean-removed stationary process.

### Ergodicity

One other property, which is primarily of conceptual value, is that of ergodicity. After forming estimates of the mean and covariance function, how can we be sure that our sample statistics reflect the true character of the underlying random field? The ergodic hypothesis states that the statistics of a random process can be derived either from repeated sampling of an ensemble of statistically equivalent media (multiple realizations) at the same point, or from samples collected at different points within a single realization of the stochastic process. Ergodicity is essentially a statement of equivalence between spatial statistics and ensemble statistics. In the real world, when we deal with spatial random processes (e.g., a geologic formation), we have only one realization and have no alternative but to adopt the ergodic hypothesis. We assume there is enough information contained in the one realization to characterize the process statistically by estimating the first two moments of its probability distribution based on a finite (usually small) number of observations of the process (see e.g., *Journal and Huijbreights* [1978]).

### Correlation Scale

Invoking the ergodic hypothesis is reasonable if the scale of the problem is much larger than the correlation scale of the process. The term “correlation scale” is often used in-

terchangeably with “correlation length” or “integral scale”. In the hydrologic literature, it represents a measure of the average distance over which the field variables are correlated, i.e., it relates information about the predictability of the process. The integral scale,  $\lambda_I$ , is defined as [Gelhar, 1984]

$$\lambda_I = \frac{1}{\sigma^2} \int_0^{\infty} C(\xi) d\xi \quad (2)$$

where  $C(\xi)$  is the covariance function,  $\xi$  is the separation distance, and  $\sigma^2$  is the variance of the process. The meaning of  $\lambda_I$  (a scalar quantity with length units) may be somewhat ambiguous because  $\lambda_I$  depends on the particular covariance function,  $C(\xi)$ . For example, the exponential covariance function,  $C(\xi) = \sigma^2 e^{-\xi/\lambda}$ , where  $\lambda$  is the correlation parameter, has integral scale

$$\lambda_I = \frac{1}{\sigma^2} \int_0^{\infty} \sigma^2 e^{-\xi/\lambda} d\xi = \lambda$$

while the Gaussian covariance function<sup>†</sup>,  $C(\xi) = \sigma^2 e^{-(\xi/\lambda)^2}$ , has integral scale

$$\lambda_I = \frac{1}{\sigma^2} \int_0^{\infty} \sigma^2 e^{-(\xi/\lambda)^2} d\xi = \frac{\sqrt{\pi}}{2} \lambda.$$

There are also covariance functions with zero integral scale; For example, a hole function defined by  $C(\xi) = \sigma^2(1 - \xi/\lambda)e^{-\xi/\lambda}$  has integral scale

$$\lambda_I = \frac{1}{\sigma^2} \int_0^{\infty} \sigma^2(1 - \xi/\lambda)e^{-\xi/\lambda} d\xi = 0.$$

In order to compare the behavior of random fields with different covariance structures, particularly for fields with zero integral scale, there needs to be another reference correlation scale that all the covariance functions can be related to. The separation  $\lambda_e$  at which the correlation function<sup>‡</sup> drops to  $e^{-1}$  is often used as a reference measure of the correlation scale. At a separation or lag distance of  $\xi = \lambda$ , both the exponential and the

---

<sup>†</sup> The Gaussian covariance function, a mathematical model for describing the spatial behavior of a process, should not be confused with the term “Gaussian process” which refers to the manner (irrespective of spatial position) in which the values of the process are distributed about the mean. In *Mantoglou and Wilson* [1981, 1982] the Gaussian covariance function is referred to as the “double exponential” covariance function.

<sup>‡</sup> The correlation function is a normalized covariance function, i.e., it is the covariance function divided by the variance and takes on values in the interval  $[-1, 1]$ .

Gaussian covariance models produce a drop in their correlation functions to  $e^{-1}$ , i.e., they both have the same correlation scale by this measure. The term “correlation scale” or “correlation length” is thus an arbitrary measure which depends on how it is defined.

Throughout this report, the term “correlation length” refers to the parameter  $\lambda$  of the covariance function rather than its formal definition given by  $\lambda_I$  or  $\lambda_e$  above. The covariance model equations which are programmed into TUBA (Table 1) are written in terms of a correlation parameter  $b \equiv 1/\lambda$ . As demonstrated in section 4.2, the correlation parameter  $b$  must be modified for the Bessel and Telis covariance models in order to affect a drop in their correlation functions to  $e^{-1}$  at the same separation as the exponential and Gaussian covariance models.

Covariance Model	Two-Dimensional Covariance Function, $C(r)$	Radial Spectral Density Function, $f(\omega)$
Exponential	$\sigma^2 e^{-br}$	$\frac{\omega/b}{b[1+(\omega/b)^2]^{3/2}}$
Gaussian	$\sigma^2 e^{-(br)^2}$	$\frac{1}{b} \left(\frac{\omega}{2b}\right) e^{-\left(\frac{\omega}{2b}\right)^2}$
Bessel	$\sigma^2 br K_1(br)$	$\frac{2(\omega/b)}{b[1+(\omega/b)^2]^2}$
Telis	$\sigma^2 \{ I_0(br) - L_0(br) + br [I_1(br) - L_{-1}(br)] \}$	$\frac{4(\omega/b)^2}{\pi b [1+(\omega/b)^2]^2}$

**Table 1.** Two-dimensional covariance models and their corresponding radial spectral density functions.  $K_1$  = modified Bessel function of the second kind of order one;  $I_0, I_1$  = modified Bessel functions of the first kind of order zero and one, respectively;  $L_0, L_{-1}$  = modified Struve functions of order zero and minus one, respectively [Abramowitz and Stegun, 1964].

### § 2.1.2 Estimation of Random Field Statistics

The discrete random fields generated by TUBA essentially represent very small sample fields taken from infinitely large, continuous theoretical random fields. Consequently, one would not expect the estimated statistics of the sample fields to match the theoretical statistics perfectly. On the other hand, one would like to have some confidence that the generation algorithm for the discrete fields preserves the desired statistical behavior.



One way of checking this would be to estimate the statistics over an ensemble of discrete fields and developing confidence intervals for, say the mean and covariance statistics, using standard statistical methods.

### Spatial versus Ensemble Statistics

In many cases, it is important to determine how well the sample statistics of a *single* realization compare with the theoretical statistics. The random field data are correlated over several correlation lengths of the process, therefore the sample field should be large relative to the correlation length of the process (spanning many correlation lengths, say 25 or more) in order to accumulate a sufficient number of independent samples upon which to compute the statistics. This spatial method of checking the statistics is valid for stationary fields and is equivalent to the ensemble method because the generation algorithm is ergodic [*Mantoglou and Wilson* , 1982]. The effect of the sample size on estimation of the mean and variance statistics is illustrated in section 4.3.1.

### Theoretical versus Sample Statistics

One may be tempted to “massage” the generated sample random field,  $Z_s(\vec{x})$ , so that its mean and variance statistics match their theoretical values exactly. For example, the transformation

$$\hat{Z}_s(\vec{x}) = \frac{\sigma_t}{\sigma_s}(Z_s(\vec{x}) - \mu_s) + \mu_t \quad \longleftarrow \quad \left( \begin{array}{l} \text{DO NOT} \\ \text{USE THIS} \end{array} \right)$$

where  $\mu_s, \mu_t$  = sample and theoretical mean respectively

$\sigma_s^2, \sigma_t^2$  = sample and theoretical variance respectively

will force the one realization,  $\hat{Z}_s(\vec{x})$ , to have  $\hat{\mu}_s = \mu_t$  and  $\hat{\sigma}_s^2 = \sigma_t^2$ . What is often overlooked however, is that  $\mu_s$  and  $\sigma_s$  are themselves random variables whose expected values are  $\mu_t$  and  $\sigma_t$  respectively. Consider only the transformation  $\hat{Z}_s(\vec{x}) = Z_s(\vec{x}) - \mu_s + \mu_t$ ; this forces the theoretical mean to be preserved for this particular realization, however, the  $\hat{Z}_s(\vec{x})$  field actually represents a sample realization drawn from a population or ensemble of random fields whose mean is given by

$$\begin{aligned} E[\hat{Z}_s(\vec{x})] &= E[Z_s(\vec{x})] - E[\mu_s] + E[\mu_t] \\ &= \mu_s - \mu_t + \mu_t \\ &= \mu_s \end{aligned}$$

Thus the one  $\hat{Z}_s(\vec{x})$  field has mean  $\mu_t$ , but that field was, in essence, drawn from an ensemble of random fields whose theoretical mean is  $\mu_s$ , not  $\mu_t$ . To make such transformations on each field that is generated, is the same as changing the target mean and

covariance statistics every time a new field is generated. This is probably not what is intended, nor is it desirable, particularly if the fields are being used in Monte Carlo simulations, therefore, this approach is not recommended. Instead, it is better to accept the fact that sample statistics based on a single realization will not match the theoretical values exactly.

In many instances, it may be of interest to observe the response of a system to a change in only the variance of the input. For example, how might the degree of reservoir heterogeneity (as reflected by the variance of the permeability field) affect the areal sweep efficiency or the production well history in a petroleum reservoir? Because different permeability fields will yield different model performance measures (even if the sample variances are equal), we seek to maintain the same pattern of spatial variability while changing only the variance (or the mean, or both) of the process. The output field can be properly scaled in both mean and variance using the transformation

$$\hat{Z}(\vec{x}) = \frac{\hat{\sigma}_t}{\sigma_t}(Z(\vec{x}) - \mu_t) + \hat{\mu}_t \quad (3)$$

where  $\mu_t, \hat{\mu}_t$  = the original theoretical and new target mean respectively

$\sigma_t^2, \hat{\sigma}_t^2$  = the original theoretical and new target variances respectively.

Here the only random quantity is the  $Z(\vec{x})$  field itself. Taking expectations,

$$\begin{aligned} E[\hat{Z}(\vec{x})] &\equiv \hat{\mu} = \hat{\mu}_t \\ E[(\hat{Z}(\vec{x}) - \hat{\mu})^2] &\equiv \hat{\sigma}^2 = \hat{\sigma}_t^2, \end{aligned}$$

shows that the target mean and variance are (in theory) preserved, although again, it is not likely that the sample statistics,  $\hat{\mu}_s$  and  $\hat{\sigma}_s^2$ , (based on a single realization) will match their theoretical (in this case, target) values.

### Variogram/Covariance Analysis

The sample field must be of sufficient resolution in order to adequately represent the covariance structure and enable it to be estimated; usually 8 to 12 output points per correlation length is sufficient for this purpose. Variogram analysis is a commonly used technique for estimating the covariance structure of a process. The variogram,  $\gamma(\xi)$ , (technically the semi-variogram) is defined by [*Journal and Huijbregts, 1978*]

$$\gamma(\xi) = \frac{1}{2} E[(Z(\vec{x} + \vec{\xi}) - Z(\vec{x}))^2]. \quad (4)$$

If we define  $\hat{Z}(\vec{\xi}) \equiv Z(\vec{x} + \vec{\xi}) - Z(\vec{x})$ , the expectation in (4) becomes  $E[(\hat{Z}(\vec{\xi}))^2]$  which is, via (1), the covariance of  $\hat{Z}(\vec{\xi})$  at zero lag ( $\xi = 0$ ) or the variance of  $\hat{Z}(\vec{\xi})$ . In

other words, the variogram is a kind of “variance of differences” between the points of the process. At zero lag, there is no variability in the differences since the points are the same, hence  $\gamma(0) = 0$ . At larger and larger lags, the amount of variability in the differences between values of the process increases to its maximum, the variance of the entire process. The variogram thus resembles an upside down covariance function and can be expressed in terms of covariances as

$$\gamma(\xi) = C(0) - C(\xi). \quad (5)$$

Variogram analysis is more powerful than covariance analysis because it is not affected by a linear drift or trend in the process while covariance estimates are. Variogram estimates for some discrete random fields generated with TUBA were calculated using (4) and are shown in Figure 12. The analysis involves fitting a theoretical variogram model (e.g., equation (5) and the covariance models in Table 1) through the data points representing the variogram estimates at each lag. Experience has shown that any one of the covariance models in Table 1 can probably be used to describe the covariance behavior of the discrete field simply by adjusting the correlation length parameter  $b$  in each covariance model. Because of this, it may be desirable to estimate the spectrum of the process, since it relates information about the spatial variation of the field that is perhaps more indicative of the “character” of the random field. The differences in the character of the fields generated using different covariance models is illustrated in Figures 8–11. Techniques for estimating the spectrum are described in *Jenkins and Watts* [1968], *Box and Jenkins* [1970], *Vanmarke* [1985], *Shumway* [1988], and others. More is said about the spectrum of the process in sections 2.3 and 3.2.

## § 2.2 The Turning Bands Method

Simulation of two and three dimensional random fields via the Turning Bands Method was originally documented by Matheron, [1973]. Presented here are the basic concepts behind the generation technique; for a more detailed description of the Turning Bands method, see *Mantoglou and Wilson*, [1981, 1982], or the review of their work in *Bras and Rodriguez-Iturbe* [1984].

The Turning Bands Method arises from purely geometrical considerations in which the multidimensional simulation is reduced to a series of unidimensional simulations taken along lines oriented at various angles as follows: From an arbitrary origin in space (the Turning Bands origin), a number of radial lines distributed at uniform spacing on  $[0, 2\pi]$  are extended outward as shown in Figure 2. The lines are evenly spaced to reduce

computation cost and insure ergodicity [Mantoglou and Wilson , 1982]. For stationary fields, a one-dimensional, mean zero, second-order stationary process,  $Z_1(\zeta)$ , is generated along each line having covariance function  $C_1(\xi)$ . Let  $Z_1(\zeta_0)$  represent the value of a line process where the orthogonal projection from the line extends through a point “o” in the simulated two-dimensional field,  $Z_s(\vec{x})$ . Then the value of the random field at that point, say at  $\vec{x} = \vec{x}_k$ , is assigned a weighted sum of the  $Z_1(\zeta_0)$  values from each line as:

$$Z_s(\vec{x}_k) = \frac{1}{\sqrt{L}} \sum_{i=1}^L [Z_1(\zeta_0)]_i \quad (6)$$

where  $L$  is the number of Turning Band lines. The algorithm is derived by applying the definition of covariance to points of  $Z_s(\vec{x})$  using equation (1) and accounting for the geometrical relationship between  $Z_1(\zeta)$  and  $Z_s(\vec{x})$ . The question remains, what is the form of  $C_1(\xi)$  and how do we generate  $Z_1(\zeta)$  ?

Mantoglou and Wilson [1982] derive the correspondence between the unidimensional covariance,  $C_1(\xi)$ , and the isotropic covariance function of the two-dimensional field,  $C_2(r)$ , and show that

$$\int_0^r \frac{C_1(\xi)}{(r^2 - \xi^2)^{1/2}} d\xi = \frac{\pi}{2} C_2(r). \quad (7)$$

Unfortunately, this is an integral equation in which  $C_1(\xi)$  can not be directly expressed as a function of  $C_2(r)$ . However, for two-dimensional isotropic processes, Mantoglou and Wilson derive a relationship between the radial spectral density function,  $f(\omega)$ , of the two-dimensional *isotropic* field and the spectral density function of the one-dimensional line process,  $S_1(\omega)$ , which results in a mathematically tractable relationship between  $C_1(\xi)$  and  $C_2(r)$ . The symbol  $\omega$  represents the spatial frequency or wave number. The two-dimensional isotropic covariance function is related to the radial spectral density function through

$$f(\omega) = \frac{\omega}{\sigma^2} \int_0^{\infty} C_2(r) J_0(\omega r) r dr \quad (8)$$

where  $J_0( )$  is a Bessel function of the first kind of order zero, and  $\sigma^2$  is the variance of the two-dimensional process. The spectrum of the line process is related to the radial spectral density function through  $S_1(\omega) = \frac{\sigma^2}{2} f(\omega)$ . Finally,  $C_1(\xi)$  can be derived from its spectral representation as

$$C_1(\xi) = 2 \int_0^{\infty} \cos(\omega \xi) S_1(\omega) d\omega = \sigma^2 \int_0^{\infty} \cos(\omega \xi) f(\omega) d\omega. \quad (9)$$

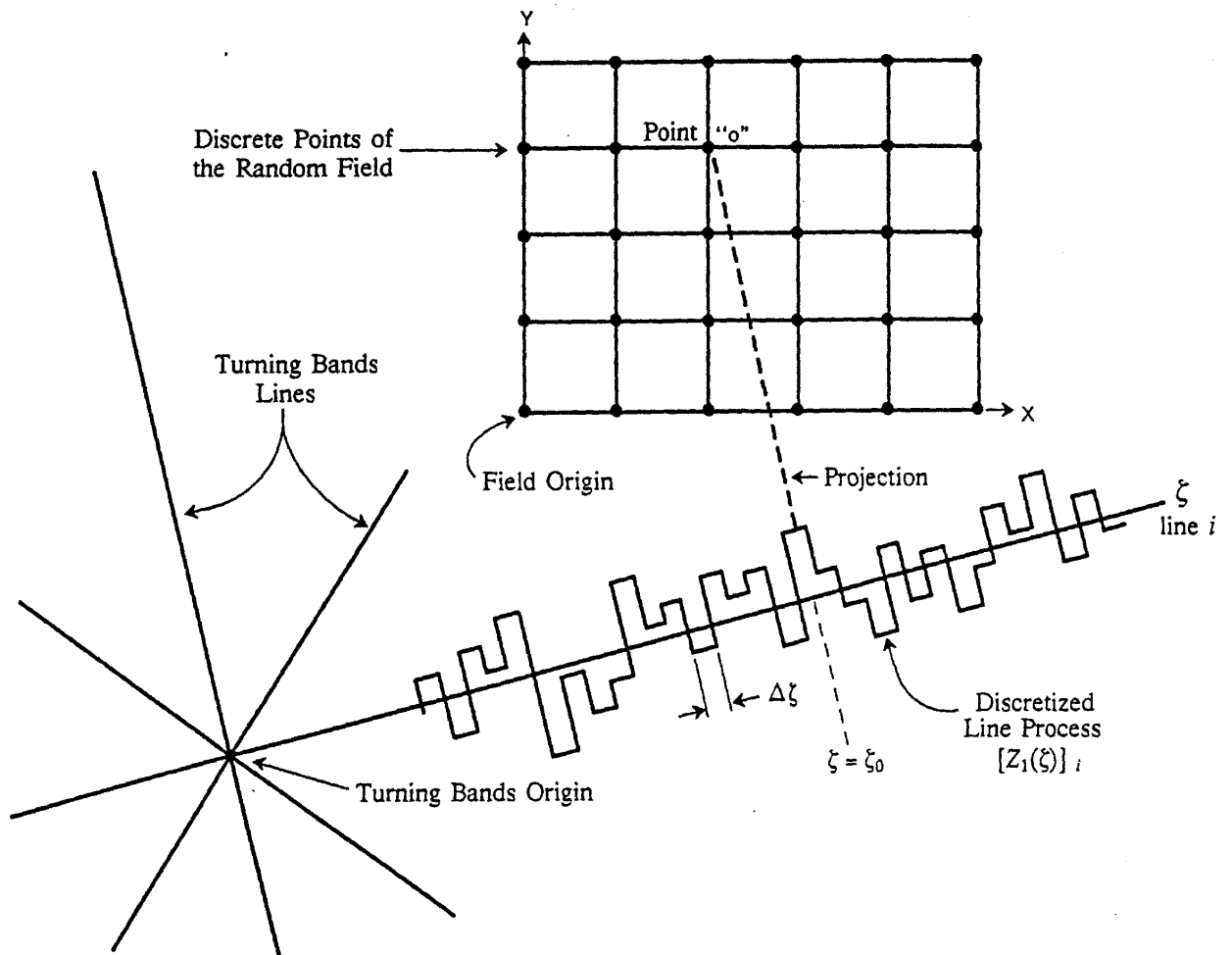


Figure 2. Schematic representation of a discrete random field and the Turning Bands lines.

Thus, given a specified two-dimensional isotropic covariance function, the corresponding line process covariance function can be derived using (8) and (9).

The line process  $Z_1(\zeta)$  can be generated using any spectral method or, depending on the properties of the unidimensional covariance function, by other techniques. The method is easily generalized to anisotropic fields. Three of the four stationary covariance models in TUBA use one of several spectral methods for generation of the line process while the fourth one uses a moving average method. The non-stationary models utilize a method proposed by *Matheron* [1973]. It represents polynomial generalized covariances using a Brownian motion process along the lines. These line process generation techniques are discussed in the following sections.

### § 2.3 Spectral Line Generation Methods

All spectral methods are founded on the spectral representation theorem which states that if  $f(x)$  is a real,  $2^{nd}$  order stationary, mean-zero stochastic process, then there exists a unique complex stochastic process,  $Z(\omega)$ , such that  $f(x)$  can be represented by the Fourier-Stieltjes integral

$$f(x) = \int_{-\infty}^{+\infty} e^{i\omega x} dZ_f(\omega). \quad (10)$$

The representation presented here is for a one dimensional process, but it can be easily generalized to multiple dimensions. The  $dZ_f(\omega)$  represent complex Fourier amplitudes of the fluctuations of the  $f(x)$  process; these are related to the spectrum of  $f(x)$  by

$$\begin{aligned} E[dZ_f(\omega)dZ_f^*(\omega')] &= 0 & \omega \neq \omega' \\ E[dZ_f(\omega)dZ_f^*(\omega')] &= S_f(\omega)d\omega = dF_f(\omega) & \omega = \omega' \end{aligned} \quad (11)$$

where \* denotes complex conjugate,  $S_f(\omega)$  is the spectral density function of  $f(x)$  and  $F_f(\omega)$  is the integrated spectrum or spectral distribution function. The spectrum (or spectral density function) of a process is a frequency domain representation which describes how the variability of the process is distributed over different spatial frequencies. Plots of the spectra and spectral distribution functions corresponding to the covariance model equations in Table 1 are shown in Figures 3 and 4 respectively. The spectral density function is related to the covariance function,  $C(\xi)$ , through the Fourier transform relationship

$$S(\omega) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} e^{-i\omega\xi} C(\xi) d\xi. \quad (12)$$

Our interest here is in generating a unidimensional process,  $Z_1(\zeta)$ , along each Turning Band line that has covariance function  $C_1(\xi)$ . We temporarily substitute  $f(x)$  for  $Z_1(\zeta)$

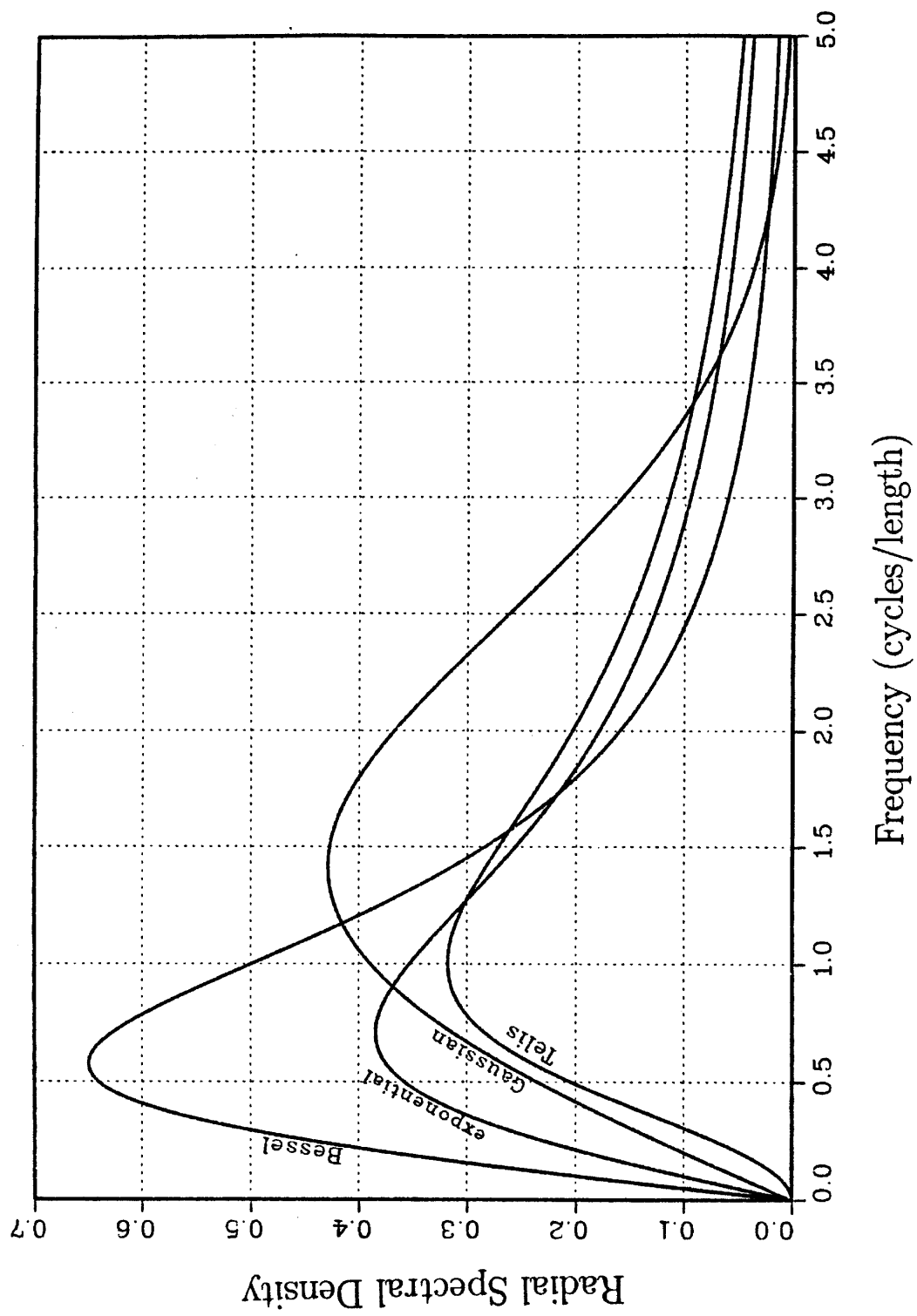


Figure 3. Radial spectral density functions for covariance models in TUBA.

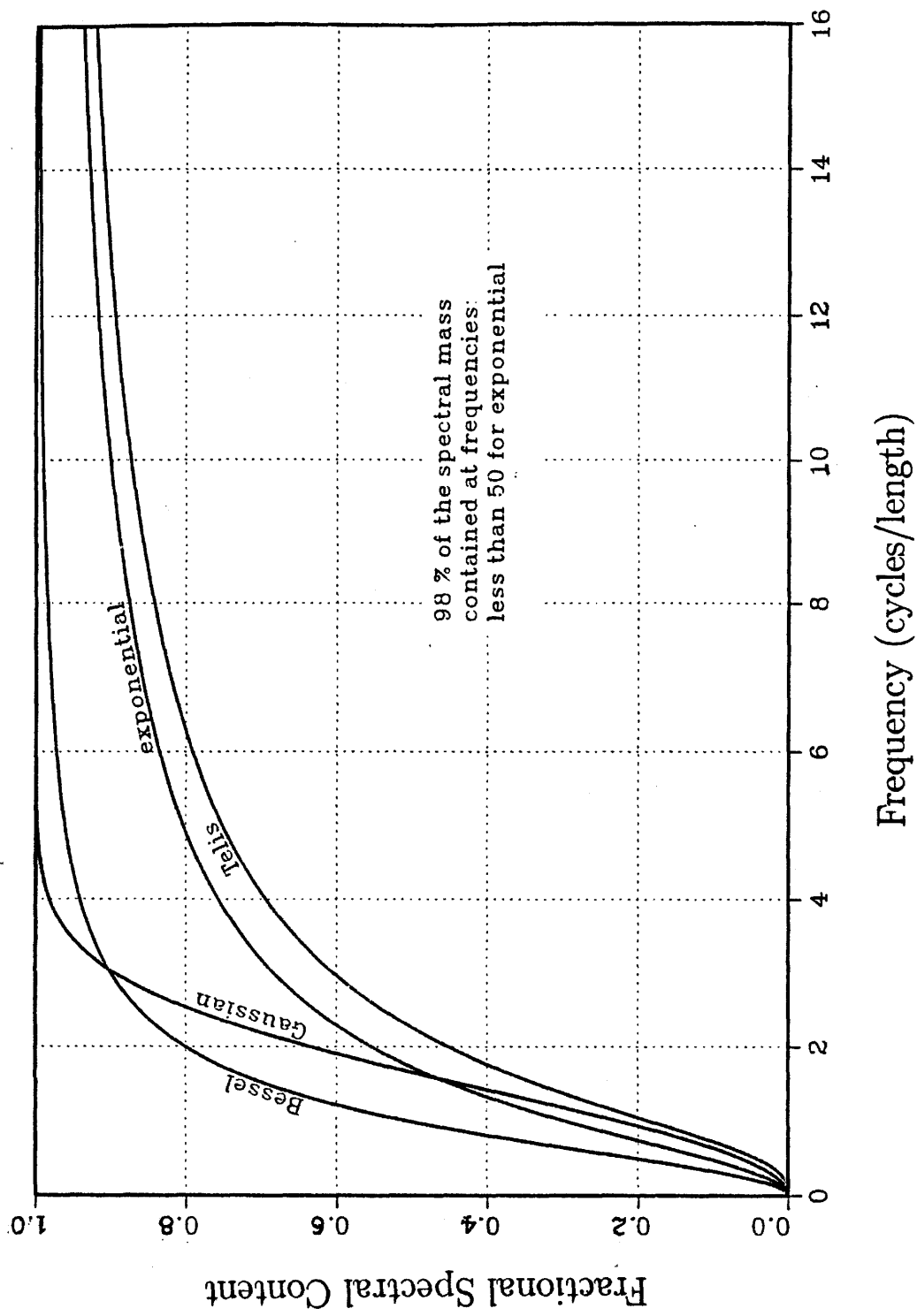


Figure 4. Spectral distribution functions for covariance models in TUBA.



in order to avoid confusion with the  $dZ_f(\omega)$  process. We use (12) to obtain the unidimensional spectral density function,  $S_1(\omega)$ , corresponding to  $C_1(\xi)$ , and then *construct* the  $dZ_f(\omega)$  process so that (11) holds. Finally, using the spectral representation theorem (10), we generate  $f(x)$  by taking the Fourier transform of the  $dZ_f(\omega)$  process.

The Fourier–Stieltjes integral (10) can alternately be written as

$$f(x) = 2\text{Re} \int_0^\infty e^{i\omega x} dZ_f(\omega) \approx 2\text{Re} \int_0^\Omega e^{i\omega x} dZ_f(\omega) \quad (13)$$

where the approximation on the right hand side of (13) results from integrating over a finite number of frequencies up to some maximum frequency,  $\Omega$ . To implement this representation on the computer, we discretize the frequency domain and replace the integral in (13) with the summation

$$f(x) \approx 2\text{Re} \sum_{j=0}^{M-1} e^{i\omega_j x} dZ_f(\omega_j)$$

where  $\omega_j = (j + \frac{1}{2})\Delta\omega$ ,  $M$  is the number of harmonics and  $\Delta\omega = \Omega/M$ . Guidelines on how to determine appropriate values for the  $M$  and  $\Omega$  parameters are presented in chapter 3. The  $dZ_f(\omega_j)$  process is constructed by setting

$$dZ_f(\omega_j) \equiv \sqrt{dF_f(\omega_j)} (U_j + iV_j)$$

where the  $U_j$  and  $V_j$  are mean-zero, variance  $\frac{1}{2}$ , uniformly or normally distributed uncorrelated random variables. Taking expectations shows that (11) is satisfied:

$$\begin{aligned} E\left[dZ_f(\omega_j)dZ_f^*(\omega_k)\right] &= 0 \quad j \neq k \\ E\left[dZ_f(\omega_j)dZ_f^*(\omega_j)\right] &= dF_f(\omega_j) E\left[U_j^2 + V_j^2\right] \\ &= dF_f(\omega_j) \approx S_f(\omega_j)\Delta\omega \end{aligned}$$

Then the  $f(x)$  process can be represented approximately as

$$f(x) \approx 2\text{Re} \sum_{j=0}^{M-1} e^{i\omega_j x} \sqrt{S_f(\omega_j)\Delta\omega} (U_j + iV_j). \quad (14)$$

The line process  $Z_1(\zeta)$  (see Figure 2) is generated using (14) with  $f \leftrightarrow Z$  and  $x \leftrightarrow \zeta$ . To take advantage of Fast Fourier Transform algorithms (FFT's), we discretize the space domain into even increments,  $\Delta x$ , and write the generation algorithm (14) in terms of the discrete points  $x_k$ , where  $x_k = k\Delta x$ ,  $k = 0, 1, \dots, M - 1$ . The Fourier

transformed function (those terms included in the summation in (14)) is a complex sequence composed of real and imaginary parts. As noted by *Tompson, et al* [1987], both the real and imaginary parts preserve the desired correlation structure; thus a single Fourier transform can be used to obtain two independent realizations of the  $f(x)$  process. In TUBA, this fact is used to advantage to reduce the computational effort by using the real part of the transformed sequence for the odd numbered lines and the imaginary part of the transformed sequence for the even numbered lines. Consequently, generation of the line process on every even numbered Turning Band line progresses quickly since the  $dZ_f(\omega)$  sequence does not need to be constructed and transformed. In addition, the correlation structure of the real and imaginary parts of the conjugate process,  $f^*(x)$ , defined by

$$f^*(x) = \int_{-\infty}^{+\infty} e^{-i\omega x} dZ_f^*(\omega), \quad (15)$$

is statistically indistinguishable from that of the  $f(x)$  process [*Tompson et al*, 1987]; because the finite Fourier transform for the conjugate process (15), is computationally cheaper than the finite Fourier transform corresponding to (10), the discrete version of (15) is used in TUBA for generating the line processes.

The spectral method of *Shinozuka and Jan* [1972] is another form of (10) that does not involve complex processes; it is also for continuous functions of space unlike the FFT method which is restricted to discrete functions of space (see also *Mantoglou and Wilson* [1982]). The Shinozuka and Jan generation algorithm is given by

$$f(x) = 2 \sum_{j=1}^M \sqrt{S_f(\omega_j) \Delta\omega} \cos(\omega'_j x + \phi_j) \quad (16)$$

$$\text{where } \omega'_j = \omega_j + \delta\omega, \quad \delta\omega = \pm\epsilon\Delta\omega, \quad \epsilon \ll 1, \quad \phi_j = U[0, 2\pi]$$

and the  $U[0, 2\pi]$  means uniformly distributed random variable on the interval  $[0, 2\pi]$ . The frequencies,  $\omega_j$ , are perturbed slightly by  $\delta\omega$  in order to reduce the tendency of the output to exhibit periodic behavior. The Shinozuka and Jan method is convenient because the output is for continuous  $x$  and because there is no restrictive connection between frequency domain and space domain parameters (see footnote p.38).

Both the FFT method and the Shinozuka and Jan method use constant frequency spacing  $\Delta\omega$ ; the frequency spacing is chosen so that rapid changes in the spectrum are adequately represented. Much of the spectrum, however, is typically very smooth and

relatively flat (see Figure 3), thus using small frequency increments throughout results in excess computational burden. This suggests that use of variable frequency spacing might be advantageous. This technique has been applied to a two-dimensional Turning Bands generator by *Munoz-Pardo and Vauchin*, [1987].

## § 2.4 The Moving Average Process and The Telis Covariance

If the line process covariance function  $C_1(\xi)$  can be written as a convolution product of a function  $f(s)$  with its transpose,  $\tilde{f}(s) \equiv f(-s)$ , then the process can be generated as a moving average process. The convolution product is defined by

$$C_1(\xi) = \int_{-\infty}^{+\infty} f(s) \tilde{f}(\xi - s) ds \quad (17)$$

where the weighting function  $f(s)$  must be found for each covariance function  $C_1(\xi)$ . In the method of moving averages, the discrete line process,  $Z_1(\zeta_i)$ , is generated from [*Journal and Huijbregts*, 1978; *Mantoglou and Wilson*, 1981]

$$Z_1(\zeta_i) = \sum_{k=-M}^M f(k\Delta s) T_{i+k} \quad (18)$$

where  $\Delta s$  is the discretization interval for the weighting function,  $f(s) = f(k\Delta s)$ , the  $T_{i+k}$  are realizations of a uniformly distributed, mean zero, uncorrelated random variable with variance  $\sigma_T^2$ , and  $M$  is chosen such that  $f(k\Delta s) \simeq 0$  for  $k \geq |M|$ . In other words, the moving average method will be practical only if  $f(s)$  dies out with increasing  $s$  and only if it dies out reasonably fast so that the summation on  $k$  is not excessive. A schematic representation of the moving average process (18) is shown in Figure 5.

The discrete covariance,  $C_d(\xi)$ , between any two points along  $Z_1(\zeta_i)$  is calculated from  $E[Z_1(\zeta_i)Z_1(\zeta_{i+\xi})]$  and is shown [e.g., *Journal and Huijbregts*, 1978] to converge to  $C_1(\xi)$ . However, a bias is introduced from the discrete approximation due to the increment size  $\Delta s$ , from truncating  $f(s)$  at some  $s_{max} = M\Delta s$ , and from  $\sigma_T^2$  being different from  $\sigma^2 = C_1(0)$ . Equivalence of variances between  $C_d(0)$  and  $C_1(0)$ , can be obtained through a simple correction [*Journal and Huijbregts*, 1978, p.536]. The other biases are discussed in Chapter 3.

The Telis covariance model (see Table 1) arises from strictly mathematical constructs associated with the moving average line method for two-dimensional Turning Bands, but it has some other interesting properties that are of practical use, especially in the field of

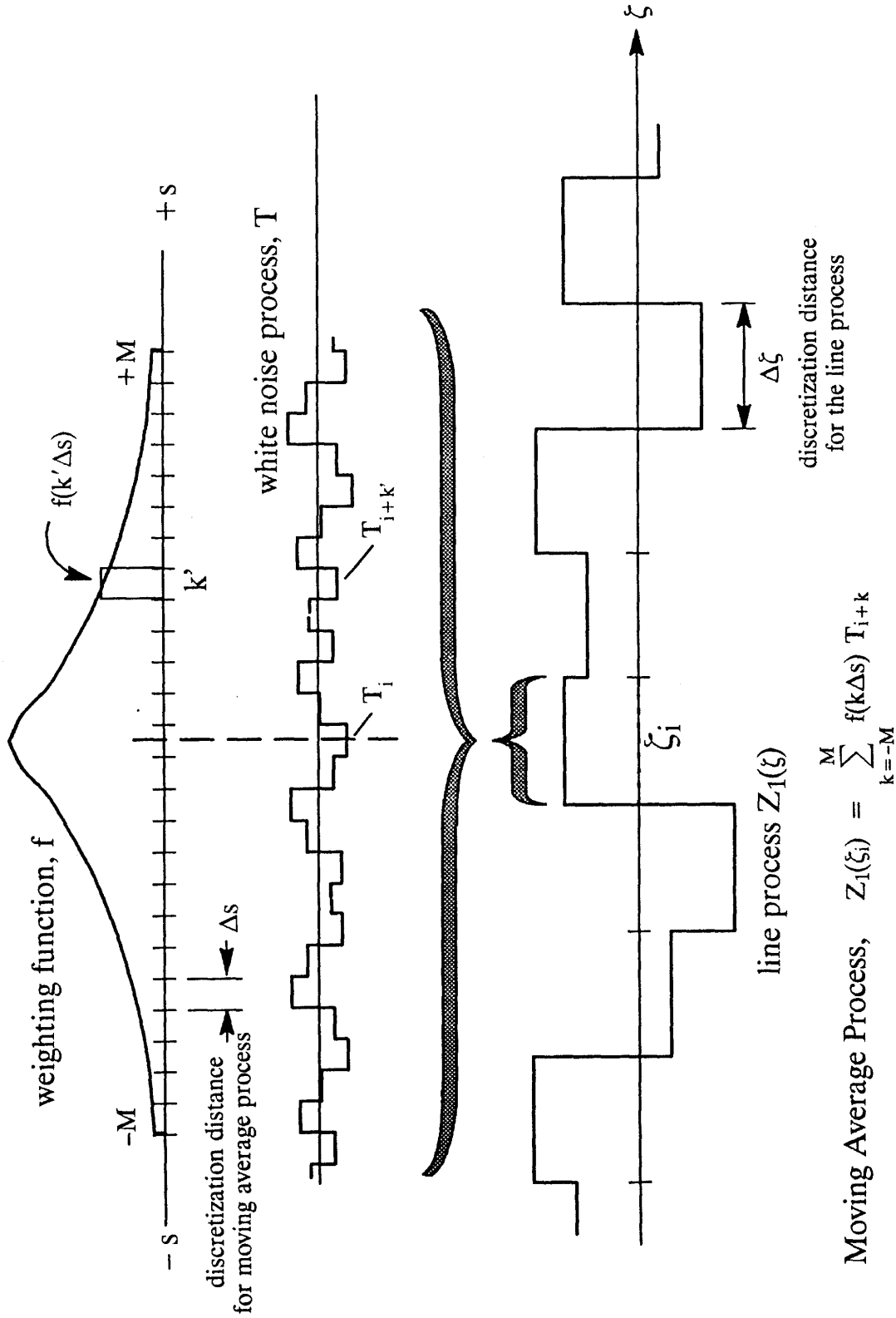


Figure 5. Schematic representation of the Moving Average process

hydrology. *Mantoglou and Wilson* [1981] examined the line process covariance functions corresponding to the two-dimensional exponential and Bessel covariance models and observed that these unidimensional covariance models closely resembled a hole function model given by

$$C_h(\xi) = \sigma^2 (1 - b\xi) e^{-b\xi}. \quad (19)$$

This unidimensional hole covariance model can easily be written as a convolution product of a function  $f(s)$  and its transpose,  $\tilde{f}(s)$ , where the weighting function,  $f(s)$ , is given by [*Journal and Huijbregts*, 1978]

$$f(s) = \begin{cases} 2\sigma\sqrt{b}(1 - bs)e^{-bs} & s \geq 0 \\ 0 & s < 0. \end{cases} \quad (20)$$

Because many two-dimensional processes exhibit covariance behavior similar to the exponential or Bessel type, they reasoned that the two-dimensional covariance function corresponding to this unidimensional hole function would also serve as a good representation of two-dimensional processes. Furthermore, through the Turning Bands transformation relating one and three-dimensional covariances [*Journal and Huijbregts*, 1978],  $C_1(\xi) = \frac{d}{d\xi}[\xi C_3(\xi)]$ , they noted that this hole function is the one-dimensional equivalent of the three-dimensional exponential covariance, i.e.,

$$C_h(\xi) = \frac{d}{d\xi}[\xi C_3(\xi)] \quad \text{where} \quad C_3(\xi) = \sigma^2 e^{-b\xi}.$$

The Telis Covariance model is the two-dimensional equivalent of the unidimensional hole function (via (7)) and therefore also corresponds to the three-dimensional exponential model. Mantoglou and Wilson theorized that since the exponential covariance function is often used to describe natural three-dimensional processes, it may be appropriate to consider the Telis function when describing two-dimensional versions of the same field resulting from spatial averaging in the remaining direction.

In hydrology, this connection between the hole function, the Telis function, and the exponential function in one, two, and three dimensions respectively, is of particular interest in problems concerning the analysis of water flow through an aquifer with spatially varying hydraulic conductivity. *Bakr et al*, [1978] found that in three dimensional flow, an exponential log-hydraulic conductivity covariance function results in a stationary piezometric head field. Similarly, their one-dimensional flow model yields stationary heads when the above hole function is used for the log-hydraulic conductivity covariance. *Mizell et al*, [1982] studied two-dimensional flows using log-hydraulic conductivity

covariance models similar in form to the Telis covariance and obtained stationary heads. This leads us naturally to pose the question: will the two-dimensional head field be stationary if the Telis function is used for the log-hydraulic conductivity covariance? More recently, *Zimmerman et al* [1987, 1988] derived the head spectrum for two-dimensional flows using the Telis covariance model and showed, by integrating the head spectrum over all frequencies, that the head field has a finite variance and is therefore stationary. Thus, the Telis covariance function is again (this time on a physical rather than a mathematical basis) the two-dimensional equivalent of the one-dimensional hole function and the three-dimensional exponential function. Because the Telis covariance function corresponds to the one-dimensional hole function, its line process covariance can be generated as a moving average process.

The moving average method described above, has not been widely used due to difficulties in deriving an analytical expression for the weighting function  $f(s)$  in (7). An alternative method for obtaining the weights has recently been proposed by *Freyberg and Black* [1987] which consists of a matrix-factorization technique for deriving the moving average coefficients. This numerical method is fast and need be performed only once for a given covariance function and density of generation points. Furthermore, only one input parameter is required and the method can be used to calculate the moving average weights for *any* specified covariance function.

## § 2.5 Generalized Covariances and Intrinsic Random Functions

When many realizations of a random field,  $Z(\vec{x})$  are available, the statistical moments (mean, variance) can be calculated at each point of the field. If the process is stationary, it means that the moments are invariant under translation and can be estimated on the basis of spatial rather than ensemble measurements (see section 2.1). If the field is not stationary, it is sometimes possible to subtract the mean,  $m(\vec{x})$ , and obtain a stationary field. For example, precipitation in an orographic region is not a stationary process due to the trend in its mean arising from changes in elevation. But many realizations of the rainfall process over time may be available to permit a reasonable estimation of the  $m(\vec{x})$  process.

In many applications, only one realization of the random field is available; for example, the distribution of ore bodies, or of aquifer or petroleum reservoir properties. In these cases, the assumption of stationarity is often made in order to calculate the

statistics of the underlying field, although this assumption is not always justified. The theory of Intrinsic Random Functions, (IRF), described in *Matheron*, [1973], was developed to deal with situations in which only one realization is available and it is not known whether the process is stationary, although it is assumed that some spatial derivative of the process is stationary. Some basic concepts of IRF theory discussed below are taken from *Delfiner*, [1978].

IRF theory is concerned with *increments* of  $Z(\vec{x})$  rather than  $Z(\vec{x})$  itself. For example, the semi-variogram  $\gamma(\vec{h}) = \frac{1}{2} E \left[ (Z(\vec{x} + \vec{h}) - Z(\vec{x}))^2 \right]$  of a constant mean process does not involve the mean because the first order difference  $Z(\vec{x} + \vec{h}) - Z(\vec{x})$  filters out constants. When  $m(\vec{x})$  is not constant, the idea is to consider higher order differences leading to the notion of generalized increments. A generalized increment of order  $k$  is defined as a linear combination  $\sum_i \lambda_i Z(\vec{x}_i)$  that filters out polynomials up to degree  $k$  (the weights  $\lambda_i$  must satisfy certain conditions). The property of being intrinsic refers to the stationarity of the generalized increments, not to  $Z(\vec{x})$  itself.  $Z(\vec{x})$  is an intrinsic random function of order  $k$  (IRF- $k$ ) if its generalized increments of order  $k$  are stationary; e.g., if  $Z(\vec{x})$  is stationary, then  $Z(\vec{x})$  is an IRF-0. An example of an IRF- $k$  is a random function of the form:

$$Z(\vec{x}) = Z_0(\vec{x}) + P_k(\vec{x})$$

where  $Z_0(\vec{x})$  is a mean zero random function with covariance function  $C(\xi)$  and  $P_k(\vec{x})$  is a  $k^{th}$  degree polynomial function with arbitrary coefficients.

Then  $Z(\vec{x})$  is classified as an IRF- $k$  with a polynomial Generalized Covariance (GC) function  $K(\xi)$ , which in this simple case equals  $C(\xi)$ . Not every function  $K(\xi)$  may serve as a Generalized Covariance, but the class of admissible functions is much broader than that of ordinary covariances. Polynomial Generalized Covariances involve only odd powers (Table 2) because they are unique only up to an even polynomial and they are isotropic because they depend only on the modulus  $|\xi|$  of the separation vector,  $\vec{\xi}$ . Polynomial GC's are very useful because they lend themselves to easy identification. *Delfiner*, [1976] believed that almost all sets of data appearing in practice can be satisfactorily described by IRF's of order 0, 1, or 2 with polynomial Generalized Covariances shown in Table 2.

TUBA has been programmed for the generation of non-stationary random fields having polynomial Generalized Covariances of the type shown in Table 2. In a manner analogous for ordinary covariances, the line process Generalized Covariance,  $K_1(\xi)$ , is

derived [Mantoglou and Wilson , 1981] and is also of polynomial type. The line process,  $Z_1(\zeta)$ , is generated using the method proposed by Matheron, [1973] which is given by

$$Z_1(\zeta) = c_0W(\zeta) + (c_1 + c_2\zeta) \int_0^\zeta W(\xi) d\xi - c_2 \int_0^\zeta \xi W(\xi) d\xi \quad (21)$$

where  $c_0, c_1$  and  $c_2$  are constants and  $W(\zeta)$  is an IRF-0 which is generated as a Weiner (Brownian motion) process. The integrals are calculated by discretizing along the line between the points where the line process is generated and numerically integrating via the trapazoidal rule. The method is fully described in *Mantoglou and Wilson* , [1981].

Order of IRF	Generalized Covariance Model	Filtered Polynomial
IRF-0	$K(h) = a_1h$	constant
IRF-1	$K(h) = a_1h + a_3h^3$	linear
IRF-2	$K(h) = a_1h + a_3h^3 + a_5h^5$	quadratic

**Table 2.** Polynomial Generalized Covariance Models in TUBA.  
After *Kafritsas and Bras*, [1981].

## § 2.6 Simulation Of Areal Average Random Fields

There are often situations in which we are interested in the areal average of a stationary point process: for example, in mining applications, we may wish to simulate average ore grade over an area; in surface water hydrology we may be interested in the average rainfall over an area; in groundwater hydrology or petroleum engineering we may wish to average aquifer or reservoir properties over the cells of a finite element or finite difference grid. The areal average process can be conceptualized by considering the process,  $Z_A(\vec{x})$  represented by

$$Z_A(\vec{x}) = \frac{1}{A} \int_{A(\vec{x})} Z(\vec{x}') d\vec{x}'$$

where  $Z_A(\vec{x})$  is the areal average process

$Z(\vec{x})$  is the point process

$A(\vec{x})$  is the averaging area referenced by the vector  $\vec{x}$

and  $A$  is the area of  $A(\vec{x})$ . If the area  $A$  and its shape are the same for all  $\vec{x}$ , *Mantoglou and Wilson* [1981] derive general expressions for  $C_A(\vec{\xi})$ , the areal average covariance



function and  $S_A(\vec{\omega})$ , the areal average spectral density function. These are given by

$$C_A(\vec{\xi}) = \int_{\mathbb{R}^2} e^{i\vec{\omega} \cdot \vec{\xi}} |H(\vec{\omega})|^2 S(\vec{\omega}) d\vec{\omega} \quad \text{and} \quad S_A(\vec{\omega}) = |H(\vec{\omega})|^2 S(\vec{\omega}) \quad (22)$$

where  $S(\vec{\omega})$  is the two-dimensional spectral density function of the point process and  $H(\vec{\omega})$  is a geometric parameter that depends on the shape of the averaging area and its relative rotation with respect to the coordinate axes. Mantoglou and Wilson derive the characteristic function  $|H(\vec{\omega})|^2$  for some typical geometries. For rectangles

$$|H_0(\vec{\omega})|^2 = \frac{16}{L_x^2 L_y^2 \omega_1^2 \omega_2^2} \sin^2\left(\frac{\omega_1 L_x}{2}\right) \sin^2\left(\frac{\omega_2 L_y}{2}\right) \quad (23)$$

where  $H_0(\vec{\omega})$  represents  $H(\vec{\omega})$  for an area with no rotation, and  $L_x$  and  $L_y$  are the dimensions of the averaging rectangle in the  $x$  and  $y$  directions respectively. This expression could be used, for example, to represent uniformly sized grid blocks of a block-centered finite difference grid. If the blocks are rotated relative to each other,  $H_0(\vec{\omega})$  is related to  $H(\vec{\omega})$  through

$$|H(\vec{\omega})|^2 = |H_0(\vec{\omega} e^{i\theta})|^2 \quad (24)$$

where  $\theta$  is the rotation angle.  $|H_0(\vec{\omega})|^2$  is also calculated for triangular averaging areas (Appendix B of *Mantoglou and Wilson*, [1981]) but is not included here because it is not programmed into TUBA. Note that even if the point process is isotropic, the areal average process will be anisotropic due to the geometric parameter  $H(\vec{\omega})$ ; the exception to this rule is when the averaging area is a circle.

The relationship between the spectra of the line processes,  $S_{1,\theta}(\omega)$ , (which depend on the line orientation angle  $\theta$ ) and the spectrum of a two-dimensional anisotropic point process,  $S(\vec{\omega})$ , is given by [*Mantoglou and Wilson*, 1981]

$$S_{1,\theta}(\omega) = \pi\omega S(\vec{\omega}) = \pi\omega S(\omega \cos\theta, \omega \sin\theta). \quad (25)$$

Thus, for areal average processes, the line process spectrum becomes

$$S_{1A,\theta}(\omega) = \pi\omega S_A(\vec{\omega}) = \pi\omega |H(\vec{\omega})|^2 S(\vec{\omega}) \quad (26)$$

where  $\theta$  is the angle between the Turning Bands line and the x-axis. Then, knowing  $H(\vec{\omega})$  and  $S(\vec{\omega})$ , the areal average process can be generated with the Turning Bands method using a spectral method for generation of the line processes.

## § 2.7 Simulation Of Anisotropic Random Fields

It was stated in section 1.2 that TUBA can generate anisotropic random fields. Except for the areal average process, the anisotropic covariance structure is limited to that of the ellipsoidal type, i.e., where the directional correlation scales are identified by a locus of lag vectors,  $\vec{\xi}$ , in an ellipsoidal shape. The anisotropic covariance,  $C(\vec{\xi})$ , is transformed to an isotropic covariance,  $C(\xi)$ , with the transformation:

$$\xi^2 = (\xi_x/\lambda_x)^2 + (\xi_y/\lambda_y)^2$$

where the  $\xi_j$ 's are the  $x$  and  $y$  direction lag distances and the  $\lambda_j$ 's are the  $x$  and  $y$  direction correlation lengths. The principal correlation scales are thus assumed to be aligned with the coordinate axes. The simulation is then performed in this transformed isotropic domain and is transformed back into the original coordinates on output. This procedure avoids the complications associated with identifying a separate covariance structure for each line process and avoids errors associated with the direct simulation of the anisotropy [*Mantoglou and Wilson* , 1981; *Tompson et al* 1987].

## Chapter 3

### Generation of the Line Processes – Practical Aspects

The purpose of this chapter is to highlight important concepts related to the accuracy and efficiency of the Turning Bands method during applications. First, geometrical considerations relevant to all the line generation methods are discussed. The type of line generation method used depends on the covariance model chosen; the line processes are generated using either *i)* a spectral method, *ii)* a moving average method, or *iii)* a Brownian motion method proposed by *Matheron* [1973]. Practical aspects associated with the computer implementation of these methods are discussed in subsequent sections. An autoregressive (AR) line process was also investigated by *Mantoglou and Wilson* [1981, p. 93]. It was the least costly method, but the resulting two-dimensional process appeared to be of little interest. Consequently, an autoregressive line process is not included in TUBA.

#### § 3.1 Geometrical Considerations

The line processes are generated as discrete processes (see illustration Figure 2) for reasons of computational efficiency: if  $N$  is the number of output points in the field, then  $N$  projections from each line will be needed for the Turning Bands algorithm. Discretizing the line process into evenly spaced increments greatly reduces the computational effort required to generate the line processes. The size of the discretization interval must be chosen so that errors introduced from the discrete approximation are small. *Mantoglou and Wilson* [1981] examined the effect of the line discretization distance,  $\Delta\zeta$ , on the accuracy of the simulation by comparing the discrete line process covariance with the theoretical one for values of  $b\Delta\zeta$  ranging from 0.05 to 4. Their analysis showed very good accuracy for  $b\Delta\zeta$  between 0.1 and 0.05. We suggest that specifying  $\Delta\zeta = 0.06/b = 0.06\lambda$  ( $\approx \frac{1}{16}$ th the correlation length) provides high accuracy without having to generate  $Z_1(\zeta)$  at an excessively large number of points along the line. An additional constraint on  $\Delta\zeta$  that should be satisfied when generating onto a gridded system is that  $\Delta\zeta \leq \Delta x$  where  $\Delta x$  is the node or cell width. This will prevent adjacent points of the random field from receiving the same projection from the Turning Band line.

## § 3.2 Spectral Generation of the Line Processes

With the exception of the moving average (Telis) and generalized non-stationary covariance models, the line processes are generated by a spectral method. The spectrum of a process is a frequency domain representation of the process which describes how the variance is distributed among different frequencies. The variance of the process is equal to the integral of the spectral density function over the entire frequency range. Space domain and frequency domain parameters are inversely related, i.e., high frequencies correspond to short range spatial fluctuations and low frequencies correspond to large or long range spatial fluctuations. Figures 3 and 4 show the radial spectral density functions and the spectral distribution functions respectively, for the covariance models in TUBA. The spectral distribution functions (or integrated spectral density functions) show how much of the spectrum is contained at frequencies less than or equal to a specified frequency. For example, the Gaussian covariance has its spectrum almost entirely contained at frequencies less than 6 cpl (cycles per unit correlation length  $\lambda = 1/b$ ).

Most spectral methods involve a numerical approximation of the spectrum, requiring the user to specify the number of harmonics ( $\mathbf{M}$ ) and the maximum frequency ( $\mathbf{FMAX}$ ) for truncation of the spectrum. If the spectrum is truncated at too low of frequency, the high frequency components of spatial fluctuations will be lost and the variance of the generated fields will fall short of the theoretical variance. Furthermore, the frequency spacing,  $\Delta k$ , must be fine enough to adequately approximate the behavior of the spectrum near the origin, otherwise, the covariance behavior at large lags will deviate from the theoretical covariance. This is an important problem when a constant frequency spacing is used because most of the rapid changes in the spectra of the covariance models (Figure 3) occur at low frequencies. When low frequency components of the spectrum are poorly represented, because  $\Delta k$  is too large, long range spatial fluctuations will not yield the large lag covariance behavior of the theoretical model. These concepts are demonstrated empirically in the following sections.

### § 3.2.1 Tests for Determining the Effect of Spectral Approximations

TUBA 2.0 uses a constant  $\Delta k$ , a finite number of harmonics,  $\mathbf{M}$ , and a finite maximum frequency,  $\mathbf{FMAX}$ , all of which introduce error into the numerical approximation of the spectrum. In order to determine the effect of the choice of the  $\mathbf{M}$  and  $\mathbf{FMAX}$  parameters on the accuracy of the simulation, we examined how well the line process covariance was preserved for different sets of  $\mathbf{M}$  and  $\mathbf{FMAX}$  input values. The accuracy of the simulations was judged on the basis of how well the variance was preserved and how well the cor-

relation structure matched the theoretical behavior. The analyses were carried out by generating multiple realizations of the unidimensional line process and calculating the line process autocovariance over the ensemble of random fields.

The covariance model chosen for this analysis was the exponential model. The one-dimensional spectral density function corresponding to the two-dimensional exponential model is given by

$$S_1(\omega) = \frac{\sigma^2}{2} \frac{\omega/b}{b[1 + \omega/b]^{3/2}} .$$

The corresponding one-dimensional covariance function,  $C_1(r)$ , derived by *Mantoglou and Wilson* [1981], is given by

$$C_1(r) = \sigma^2 \left\{ 1 - \frac{\pi}{2} br [I_0(br) - L_0(br)] \right\}$$

where  $I_0$  and  $L_0$  are modified Bessel and modified Struve functions respectively, of order zero. The theoretical  $C_1(r)$  function was evaluated using equation 12.2.3 of *Abramowitz and Stegun* [1964] and the Longman method [Longman, 1956] was used for evaluating the infinite integral in that equation.

The simulations were designed so that the statistical analysis of each run was based on random field data spanning 10,000 correlation lengths. The theoretical variance was set equal to 1.0. Results of the sensitivity analyses are discussed in the following sections.

### § 3.2.2 Effect Of Truncating The Spectrum

It was noted in earlier that the variance is equal to the integral of the spectrum; i.e., the integrated spectrum (Figure 4) tells you how much of the variance you can expect to recover when the spectrum is truncated at some finite frequency. For example, Figure 4 shows that, for the exponential model, one can expect to recover 90% of the variance if the spectrum is truncated at a frequency of 10 cpl. This was verified by running five cases where the maximum frequency, **FMAX**, was set at 5, 10, 20, 50, or 100 cpl. In each case,  $\Delta k$  was kept very small to avoid errors associated with coarse frequency spacing. The input parameters and output field variances are shown in Table 3. The theoretical and sample statistics show excellent agreement. Figure 4 can thus be used as a guide for deciding where to truncate the spectrum for the Bessel and Gaussian models as well. An example of the effect that premature truncation of the spectrum can have on the character of the random field is illustrated in section 4.3.3.

Maximum Frequency FMAX	Number of Harmonics M	Frequency Spacing $\Delta k$	Spectral Content %	Sample Variance $\sigma^2$
5.	128	0.04	80	0.8071
10.	256	0.04	90	0.8963
20.	512	0.04	95	0.9577
50.	1024	0.05	98	0.9759
100.	2048	0.05	99	0.9982

Table 3. The effect of truncating the spectrum at finite frequencies on the variance of the random fields. (Theoretical variance = 1.0)

### § 3.2.3 Effect of Discretizing the Spectrum

Based on the theoretical and experimental results (Table 3) of the previous section, the choice of FMAX=100 was deemed sufficient for preserving the variance of the process. In these experiments, FMAX was maintained at 100 cpl while the frequency spacing,  $\Delta k$ , varied from 0.05 to 1.0 cpl. The results from these runs are plotted in Figure 6 and demonstrate how the large lag covariance behavior deviates from the theoretical model when  $\Delta k$  is too large. Based on these results and the graphs in Figure 4, it is suggested that  $\Delta k=0.1$  or less is sufficiently small for the Gaussian model, while  $\Delta k=0.02$  or less may be necessary for the Bessel model.

These results are summarized in Table 4. The M and FMAX parameters for the Shinzuka and Jan method (Table 4b) are set at their minimum required values in accordance with the preceding sensitivity analysis; the FFT parameters (Table 4a) must be specified such that the proper  $\Delta k$  and the minimum  $\Delta \zeta$  are maintained (see section 3.1 and the footnote at the bottom of page 38).

Turning Band parameter	exponential model	Gaussian model	Bessel model
FMAX	100	100	100
$\Delta\zeta$	$0.06\lambda$	$0.06\lambda$	$0.06\lambda$
M	2048	1024	4096
$\Delta k$	0.05	0.10	0.02

Table 4a. Spectral parameters for generating the line processes via the Fast Fourier Transform method.

Turning Band parameter	exponential model	Gaussian model	Bessel model
FMAX	100	6	10
$\Delta\zeta$	arbitrary	arbitrary	arbitrary
M	2000	60	500
$\Delta k$	0.05	0.10	0.02

Table 4b. Spectral parameters for generating the line processes via the method of Shinozuka and Jan.

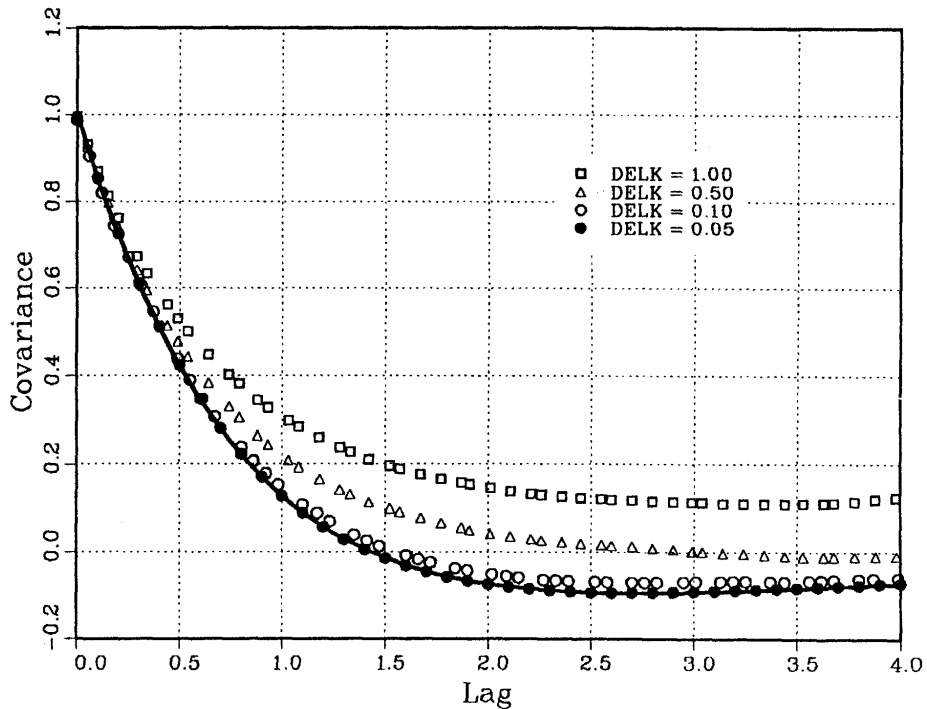


Figure 6. The effect of frequency spacing,  $\Delta k$ , on the correlation structure of the line process. Solid line is the theoretical covariance function for the exponential model.

### § 3.3 Moving Average Generation of the Line Process

The moving average method was discussed in section 2.4 and is illustrated in Figure 5. Assuming that the line process discretization distance  $\Delta\zeta$  is chosen sufficiently small to adequately represent the line process covariance behavior (section 3.1), the question that remains is how small to make  $\Delta s$  and how big to make  $M$ ? The weighting function  $f(s)$  (equation (20)) is a decaying exponential function which is, for all practical purposes, zero for  $s \geq 5/b = 5\lambda$  for the hole function equivalent to the Telis covariance function (see Figure 7) The appropriate choice for  $\Delta s$  was determined by: choosing a  $\Delta s$ , generating a multitude of random fields (the 1D line process), calculating the autocovariance over the ensemble of realizations, and comparing that result with the theoretical covariance (equation (19)); this process was repeated using different  $\Delta s$ 's until the discrete covariance matched the theoretical covariance. Based on the results of these tests (shown in Figure 7), it is seen that the discretization interval for the hole function moving average process should be approximately  $0.05/b = 0.05\lambda$  or less, or about 1/20th of the correlation length.

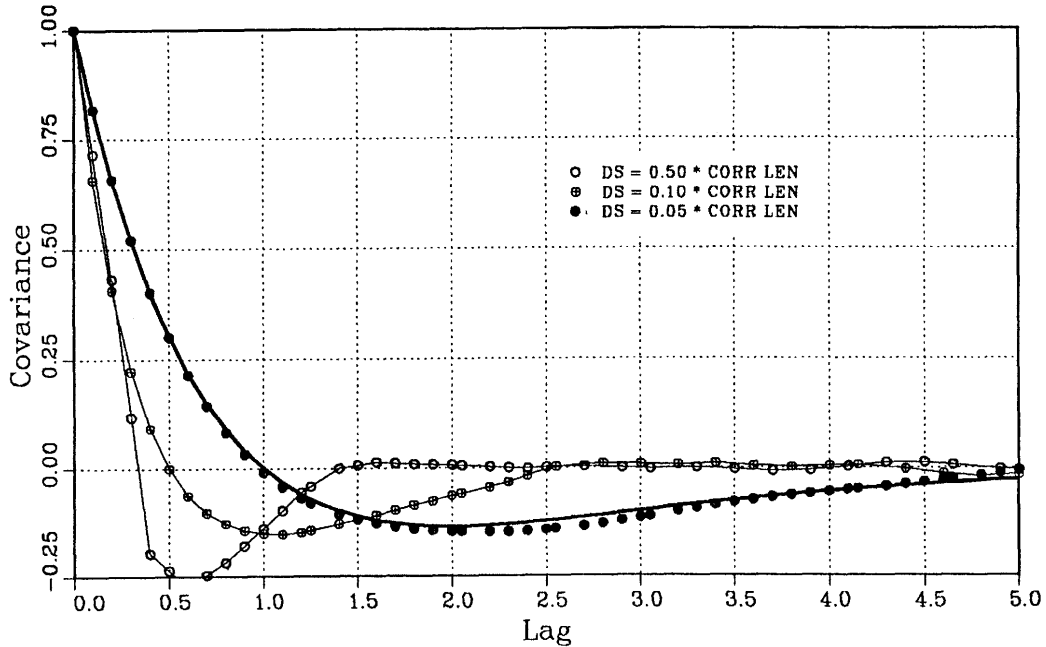


Figure 7. Effect of discretization interval  $DS$  on the correlation structure of the line process. Thick solid line is the hole function covariance model, Equation (19).



## Chapter 4

### User's Manual

This chapter focuses on implementation and application of the TUBA computer program. The objectives of this chapter are to discuss what and how input data to the code are handled, to provide general guidelines to assist the user in selecting a consistent set of input values, and to illustrate how to run the program. Numerous examples are given to demonstrate the versatility of the code and to illustrate the variability in the character of the generated random fields. Included are some examples relating to the discussion on estimating random field statistics (section 2.1.2) and on the practical aspects of line generation (chapter 3).

#### § 4.1 Code Input Description

Input to TUBA is handled by querying for *i*) output field parameters, *ii*) covariance model parameters, *iii*) Turning Band parameters, *iv*) output file parameters, and *v*) simulation parameters in that order. Field parameters refer to the geometry and resolution of the field being simulated. Covariance model parameters specify the choice of covariance model to be used, the desired mean and variance, etc. Turning Band parameters refer to control parameters for the Turning Bands method such as the number of Turning Band lines. Output file parameters control how the data is written to the output files. Simulation parameters determine how many realizations will be produced etc. Guidelines on how to choose these input parameters are presented in the following sections.

##### § 4.1.1 Output Field Parameters

The random fields may be generated at arbitrary points in space, such as the nodal points or Gauss points of a finite element grid, or onto the grid points of a regularly-spaced or irregularly-spaced finite difference grid. The grid may be a point or a block centered grid, the difference being whether the coordinate origin is at the center of the first nodal block (cell) or at the lower left hand corner of the cell. For gridded output, an optional “mask file” can be used; the mask file is a matrix (the same size as the output grid) containing zero and non-zero values. Typically, only 1's and 0's are used, although any zero and non-zero values will work. An exact zero tells TUBA to skip the random field calculations for that grid point, while a non-zero value means yes, generate a random field value here (see section 4.3.2).

If generating at arbitrary points in space, TUBA must read the  $\vec{x} = (x, y)$  coordinate locations at which to generate values of the random field. TUBA asks for the name of the file containing the  $(x, y)$  data and reads the coordinate pairs using a free format read statement (`READ(LU,*)`). It is best to simply place one  $(x, y)$  coordinate pair per line in the input file. TUBA automatically counts the number of pairs read and reports this number to the screen.

If generating onto a uniform grid, TUBA asks for the dimensions of the rectangular region and the number of nodes or cells in the  $x$  and  $y$  directions. TUBA uses the output field dimensions (length units) and the grid dimensions (nodes or cells) to calculate the grid spacing,  $\Delta x$ . The lower left hand corner of the output field (in plan view) is placed at the origin of the coordinate system. Thus, for point centered grids the coordinates of the first node will be  $(0., 0.)$  whereas for block centered grids it will be  $(\frac{\Delta x}{2}, \frac{\Delta y}{2})$ .

In order to adequately preserve the desired statistics (mean, variance) in a single realization, the size of the field should be large relative to the correlation scale (see section 2.1.2), on the order of 20 or more correlation lengths. A common mistake is to generate fields with only a few correlation lengths, obtain biased statistics, and conclude that there is a coding error. The bias occurs because the sample size is too small for meaningful single-realization statistical analysis. This is clearly demonstrated by generating hundreds of these small domains, and accumulating the statistics over them; the ensemble statistics will preserve the desired target statistics.

The grid spacing,  $\Delta x$ , should also be small relative to the correlation length, or the resulting generated process will be smoother than the true process, and statistical tests will show that the higher frequency components will not be preserved.

The random field,  $f(\vec{x})$ , can be generated as a normally or log-normally distributed process; i.e., TUBA provides the option of exponentiating the generated field as  $10^{f(\vec{x})}$  or  $\exp\{f(\vec{x})\}$  to generate the log-normal field. Other user desired transformations can be handled via post-processing.

### § 4.1.2 Covariance Model Parameters

TUBA asks the user to choose from a selection of five covariance functions: the *i*) exponential, *ii*) Gaussian, *iii*) Bessel, *iv*) Telis, and *v*) Generalized non-stationary covariance models. In addition, a user-specified stationary covariance model can be input (see section 5.4). Covariance plots for the first four of these models are shown in Figure 1. For stationary fields, TUBA queries for the desired mean and variance of the random field; these values must always be specified for the normally distributed process. For example,

if you wish to generate a log-normal process, e.g., a hydraulic conductivity field with a mean of 100 millidarcies, then you would input the desired mean equal to 2 and specify base 10 exponentiation.

Each stationary covariance model has two correlation scale parameters, the  $x$  and  $y$  correlation lengths which represent the average distance over which the field variables are significantly correlated in the  $x$  and  $y$  directions. Recall that (section 2.1.1) the “correlation length” refers to the parameter  $\lambda = 1/b$  in the covariance model equations (Table 1). The correlation lengths should be specified such that there is sufficient resolution to capture the covariance behavior of the process; we recommend using at least 10 output points per correlation length (see section 4.3.2). To generate isotropic random fields the  $x$  and  $y$  direction correlation lengths should be equal; for an  $x:y = 5:1$  anisotropic correlation structure, the  $x$ -direction correlation length should be five times larger than the  $y$ -direction correlation length and so on.

For non-stationary fields, TUBA queries for the coefficients to the generalized covariance function  $K(r) = a_1r + a_3r^3 + a_5r^5$ . Generalized covariances are discussed in section 2.5 and example fields are shown later in this chapter.

### § 4.1.3 Turning Band Parameters

Turning Band parameters control the way in which the line processes are generated. If the line processes are not properly generated, the two-dimensional field will not exhibit the desired covariance behavior. Carefully following the guidelines presented here should result in random fields possessing the proper statistical behavior.

TUBA has a convenient option which relieves the user of the burden of figuring out what values to input for these parameters; i.e., the Turning Band parameters can be calculated internally and automatically by TUBA. It may be best to use this default option rather than risk inputting inappropriate values (see section 4.3.5). The remainder of this section concerns the manual entry of the Turning Band parameters.

TUBA queries the user for the number of the Turning Bands lines. Based on a sensitivity analysis of covariance convergence to the number of Turning Bands lines described in *Mantoglou and Wilson, [1981]*, we recommend using at least 16 lines.

TUBA queries for the line process discretization distance. Because the line processes are discrete processes, the discretization distance along the lines must be specified and must be chosen so that errors introduced from the discrete approximation are small. *Mantoglou and Wilson [1981, 1982]* examined the effect of the discretization distance,  $\Delta\zeta$ , along the Turning Band lines on the accuracy of the simulation and concluded

that  $\Delta\zeta$  should be no larger than  $0.1\lambda$  where  $\lambda$  is the correlation length. Based on the analyses in that report and our own experience, we recommend using  $\Delta\zeta = 0.06\lambda$  (approximately 16 points/correlation length).

For the exponential, Gaussian, and Bessel covariance functions, the line processes are generated using a spectral method; if the default Turning Band parameters are used, the line processes are generated using the Fast Fourier Transform (FFT) method, otherwise the much slower method of *Shinozuka and Jan* [1972] is used. This is because certain parameters would be overspecified if the FFT method were used<sup>†</sup>. Two parameters control the way in which the spectrum is approximated: these are **FMAX**=the maximum frequency at which the spectrum is truncated, and **M**=the number of harmonics into which the spectrum is discretized. The constant frequency spacing,  $\Delta k$ , is determined by  $\Delta k = \text{FMAX}/\text{M}$ . The recommended minimum **M** and **FMAX** values for each covariance model is shown in Table 4 which was developed from the sensitivity analyses discussed in section 3.2.

If the Telis covariance model is chosen, the line processes are generated via a moving average method. Here the discretization distance,  $\Delta s$ , for the weighting function  $f(s)$  and the white noise process  $T$  is needed (see Figure 5). Based on the analysis described in section 3.3, this discretization distance should be set no larger than  $\Delta s = 0.05\lambda$ .

If a user specified covariance model is input, the user will have to determine the optimum **M** and **FMAX** parameters for the corresponding spectral density function if a spectral method is used to generate the line processes or the optimum  $\Delta s$  value if a moving average method is used to generate the line processes.

Finally, for non-stationary intrinsic random fields of order one or two, the discretization distance,  $\Delta W$ , for the Weiner process is required. This discretization interval is needed for numerically approximating the integrals shown in equation (21). The size of  $\Delta W$  required for proper generation on the intrinsic line process was not determined in the same manner that the discretization parameters for the other line generation methods were found. When the default Turning Band parameters are chosen, TUBA sets  $\Delta W = 0.2\Delta\zeta$  where  $\Delta\zeta$  is the Turning Bands line discretization distance.

---

<sup>†</sup> With the FFT method, the line process discretization interval,  $\Delta\zeta$ , is related to the maximum frequency, **FMAX**, by  $\Delta\zeta \equiv 2\pi/\text{FMAX}$ . When the Turning Band parameters are entered manually,  $\Delta\zeta$  and **FMAX** are set independently. With the method of Shinozuka and Jan, there is no connection between  $\Delta\zeta$  and **FMAX**, thus, these parameters can be set independently. Under certain conditions, however, the FFT method *will* be used when entering the Turning Band parameters manually; see the footnote under section 4.2.4.

#### § 4.1.4 Output File Parameters

TUBA asks for the filename for writing the data to; if multiple realizations are being generated, each output field is written to a separate file, the file extension is stripped off and replaced with the simulation number. For example, if the name `FIELD.DAT` is given and three realizations are to be generated, the output files will be named `FIELD.1`, `FIELD.2`, and `FIELD.3`. Alternatively, all realizations may be written to a single output file.

The data may be written out formatted or unformatted—unformatted is recommended for large output files because the I/O is faster, unformatted data files take up less space on the disk, and “manual data analysis” (looking at a file full of numbers) is generally not an efficient way to examine the data. Variogram analysis, spectral analysis, or simply plotting the data as a contour or shaded relief map are more powerful methods for examining the behavior of the random fields.

If you are generating field values at arbitrary locations in space, the output may be configured to print only the field values (i.e., to exclude reflecting the coordinate locations which already reside in a file that is read as input). For gridded output, TUBA asks whether the data should be written out “with a single write statement” or “one line at a time.” The single write statement option means the entire random field is written using one Fortran `WRITE` statement. For formatted output, we recommend using the “one line at a time” option; for unformatted output, either option is acceptable, depending on how the data are to be read by other post-processing programs.

If the gridded data are to be formatted and written out line by line, there is an option to have it write out the rows beginning with the last row and ending with the first; this way the data will appear on the page “in plan view”, i.e., with the origin at the lower left and the last row on top. This option is intended for use with small data sets where the entire grid will fit onto a single page. An example where this option is invoked is shown in section 4.2.5.

#### § 4.1.5 Simulation Parameters

Multiple realizations of the random field can be generated as desired; TUBA queries for the number of realizations to be simulated. If more than one realization is produced, the data will be written to the output file or files using the same format each time.

Two different random number generators have been included in TUBA: For most applications, we recommend using the algorithm of Marsaglia and Bray [*Dudewicz and Rally*, 1981]. However, all of the examples in this manual were run using the other

generator [Swain and Swain, 1980] because it is a *machine independent* random number generator and thus will allow the results shown here to be duplicated on different machines. Both generators need an integer seed (positive number) to initialize the sequence of random numbers. For the Swain and Swain generator, the seed must be composed of eight digits. No such restriction is imposed by the Marsaglia and Bray generator. The latter generator involves multiplications of large integers which results in integer overflow conditions (the product is a number which is too large to store in an integer word). For this reason, TUBA must be compiled with the integer overflow check turned off (depending on the Fortran compiler, see section 5.3).

## § 4.2 Example Runs – Input, Output, and Analysis

Sample problems which exercise all of the options in TUBA are illustrated in the following sections. The objective of this section is to show how to set up and run the program in batch mode and to illustrate what the user will see when running TUBA interactively at the computer terminal. The output files are analysed by computing their sample statistics (mean and variance), producing shaded contour maps, and by calculating variograms or estimating their spectral properties.

Because the sequence of questions is identical for many of these examples, the interactive session is illustrated only once, for the most common case of generating a stationary random field onto a regular grid. In (almost) all of the other cases, the input values used to generate the fields are shown. With a few exceptions, all of these runs are made by using the default Turning Bands parameters. In addition, all of these runs are made with the *machine independent* random number generator to enable the user to duplicate these results within the limits of machine round off error.

During execution of the program, TUBA reports certain information to the screen concerning its progress in the sequence of computations; this is useful when running TUBA interactively. Prior to generating the line processes, TUBA must calculate the projections from every point in the field onto each Turning Band line; this task constitutes a significant portion of the overall computation time and therefore its computation progress is listed<sup>†</sup>.

Also prior to the line generation computations, certain line process array data is calculated when the spectral method is used to generate the line processes. These

---

<sup>†</sup> For very large fields (say greater than 500,000 nodes) it may be better to generate multiple realizations in one run if more than one realization will eventually be needed, since the projection calculations need only be done once in a multiple simulation run.

calculations proceed rapidly (except when areal averaging is used) but computation progress is listed anyway in order to show the number of harmonics used along each Turning Band line. The number of harmonics listed may differ from the values shown in Table 3 in order to satisfy the constraints of a particular simulation (see section 4.2.6). During generation of the line processes, progress as to which Turning Band line is being generated is reported to the screen. If the FFT spectral method is used to generate the line processes, the even numbered lines will be generated much faster than the odd numbered lines; the reason for this is explained in section 2.3.

### Reflection of Input Parameters

Because TUBA was written primarily for interactive use, it does not ask for input parameters it will not need, i.e., the number of “cards” (input lines of control variables) will vary from simulation to simulation depending on which options are invoked. Therefore, as a convenience to the user, TUBA automatically generates an input file “on the fly” that can later be used for batch processing. The file has the same name as the output data file but a different extension, e.g., if the output file is named `FIELD.DAT`, the newly created input file is named `FIELD.INP`. This provides a convenient means of keeping an accurate record of the input and establishing a “template” control file that can be edited and used for subsequent batch or interactive runs. On operating systems that use redirection symbols (`<`, `>`), TUBA can be run in “batch-interactive” mode by typing “`TUBA < FIELD.INP`” at the command prompt; TUBA will then read all input data from the file `FIELD.INP` instead of the keyboard.

### Reflection of Turning Band Parameters

In addition to listing the input parameters used to create the field, this file lists out some internal parameters which may be important for regenerating the field or portions of it (see section 4.2.6), and lists the sample statistics of the realization(s). The sample statistics are always calculated *before* exponentiation occurs (in cases where log-normal fields are generated). Ensemble statistics are also listed when multiple realizations are generated. The appearance of the fields and their sample statistics will be affected by the size and resolution of the grid onto which they are generated. Therefore, TUBA also lists out (for the stationary models only) the number of output points/correlation length used (as determined by the geometrical input variables) and an approximation of the number of independent samples the the field represents. This may help to provide a clue as to why the sample statistics deviate from their theoretical values (see sections 2.1 and 4.3.1).

### Comparison of Gridded Fields

In the examples that follow, these <NAME>.INP files are shown exactly as they appear on output (for TUBA version 2.0). Incidentally, all input data (real or integer) are read with a free format (`READ(LU,*)`) read statement, so that no formatting of input values into specific columns of the batch input file is necessary.

In the examples involving gridded output, the size of the output grid is 100 x 100 nodes. For stationary fields, the correlation lengths are specified so that all of the correlation functions drop to  $e^{-1}$  at the same lag or separation. This was done to better enable a visual comparison of the correlative properties of the fields generated with these different covariance models (see section 2.1.1). At a lag distance of  $\xi = \lambda = \lambda_e$ , both the exponential,  $\rho_e(\xi)$ , and Gaussian,  $\rho_G(\xi)$ , correlation functions drop to  $e^{-1}$ . For the Bessel,  $\rho_B(\xi)$ , and Telis,  $\rho_T(\xi)$ , correlation functions, the separation distances at which this condition is reached is  $1.65\lambda$  and  $0.75\lambda$  respectively. Thus, if the correlation length for the exponential and Gaussian models is specified as  $\lambda$ , the correlation lengths for the Bessel and Telis models must be given as  $\frac{\lambda}{1.65}$  and  $\frac{\lambda}{0.75}$  respectively, in order that  $\rho_B(\frac{\lambda}{1.65}) = \rho_T(\frac{\lambda}{0.75}) = \rho_e(\lambda) = \rho_G(\lambda) = e^{-1}$ .

Shaded contour maps of the generated random fields were produced in order to illustrate the character of the spatial variability patterns. These maps were generated by normalizing the field values into “levels” corresponding to their relative deviation from the sample mean and assigning a shade pattern to each level (dark = low (below mean) value, white = high (above mean) value). The plots show eight shading levels (equally spaced from  $-3$  to  $+3$  standard deviations about the mean). Although the generated fields appear to be discrete, because of the stair steps of the shade patterns used here, they are in fact continuous. Smooth contouring of previous TUBA fields are shown in *Mantoglou and Wilson* [1981, 1982].

#### **§ 4.2.1 Generation of Stationary Isotropic Random Fields**

Random fields corresponding to each of the four stationary covariance functions shown in Figure 1 and Table 1 are generated in this section. The interactive session is illustrated only for the Gaussian covariance model; the contents of the <NAME>.INP output files showing the input parameters used precedes the plot of each field. The user’s responses to the questions are reflected back to the screen (preceded by `****`) in order to verify the input, particularly for batch runs. The generated fields are plotted in Figures 8 through 11 and variogram estimates of these fields are plotted in Figure 12. These results are discussed at the end of this section.



## Interactive Session For Gaussian Covariance Model

+++++++ Program "TUBA" ++++++

A Code For Simulating 2D Random Fields  
Via The Turning Bands Method

+++++++ OUTPUT FIELD PARAMETERS ++++++

(1) - Simulate Only At Specified (x,y) Coordinates  
(2) - Simulate Onto The Nodes Of A Rectangular Grid  
\*\*\*\*\*  
2

(1) - Point Centered Grid  
(2) - Block Centered Grid  
\*\*\*\*\*  
2

Enter The Maximum X And Y Field Dimensions

\*\*\*\*\* 100.0000 100.0000

Enter The Number Of Nodes In The X And Y Directions

\*\*\*\*\* 100 100

Enter The Mask Filename or Type NONE

\*\*\*\*\* NONE

(1) - Generate a Field  $f(x)$  Whose pdf is Normal  
(2) - Generate a Lognormal Field  $K(x) = \exp(f(x))$   
(3) - Generate a Lognormal Field  $K(x) = 10^{f(x)}$   
\*\*\*\*\*  
1

+++++++ COVARIANCE PARAMETERS ++++++

Select Type Of Covariance Model:

(0) - User Specified  
(1) - Exponential Model  
(2) - Gaussian Covariance  
(3) - Bessel Type Covariance  
(4) - Telis Covariance Function  
(5) - Generalized Covariance Model  
\*\*\*\*\*  
2

Enter Desired Mean And Variance

\*\*\*\*\* 0.000000E+00 1.000000

Enter The X and Y Direction Correlation Lengths

Make These Equal For Isotropic Fields

\*\*\*\*\* 20.00000 20.00000

+++++++ TURNING BANDS PARAMETERS ++++++

(1) - Use Default Turning Band Parameters  
(2) - Enter The TBM Parameters Manually  
\*\*\*\*\*  
1

+++++++ OUTPUT FILE PARAMETERS ++++++

Enter A Filename For The Output File(s)

\*\*\*\*\* GAUSS1.DAT

- (1) - Unformatted Output
- (2) - Formatted Output

\*\*\*\*\* 1

- (1) - Write Out Matrix With One WRITE Statement
- (2) - Write Out Matrix One Line (Row) At A Time

\*\*\*\*\* 2

+++++++ SIMULATION PARAMETERS ++++++

- (1) - Marsaglia and Bray Random Number Generator
- (2) - Machine Independent Random Number Generator

\*\*\*\*\* 2

Enter An Integer Seed To Initialize The Generator  
Seed For This Generator Must Be 8 Digits Long

\*\*\*\*\* 76651594

Enter The Number Of Realizations To Be Simulated

\*\*\*\*\* 1

Number Of Elements Allocated In A Array = 4999000  
Total Storage Required For Computations = 208945

Calculating Projections ... Point No	500 ... ( 5.0 % )
Calculating Projections ... Point No	1000 ... ( 10.0 % )
Calculating Projections ... Point No	1500 ... ( 15.0 % )
Calculating Projections ... Point No	2000 ... ( 20.0 % )
Calculating Projections ... Point No	2500 ... ( 25.0 % )
Calculating Projections ... Point No	3000 ... ( 30.0 % )
Calculating Projections ... Point No	3500 ... ( 35.0 % )
Calculating Projections ... Point No	4000 ... ( 40.0 % )
Calculating Projections ... Point No	4500 ... ( 45.0 % )
Calculating Projections ... Point No	5000 ... ( 50.0 % )
Calculating Projections ... Point No	5500 ... ( 55.0 % )
Calculating Projections ... Point No	6000 ... ( 60.0 % )
Calculating Projections ... Point No	6500 ... ( 65.0 % )
Calculating Projections ... Point No	7000 ... ( 70.0 % )
Calculating Projections ... Point No	7500 ... ( 75.0 % )
Calculating Projections ... Point No	8000 ... ( 80.0 % )
Calculating Projections ... Point No	8500 ... ( 85.0 % )
Calculating Projections ... Point No	9000 ... ( 90.0 % )
Calculating Projections ... Point No	9500 ... ( 95.0 % )
Calculating Projections ... Point No	10000 ... ( 99.9 % )

Line Process Array Data ... Harmonic	103 ... ( 5.0 % )
Line Process Array Data ... Harmonic	205 ... ( 10.0 % )
Line Process Array Data ... Harmonic	308 ... ( 15.0 % )
Line Process Array Data ... Harmonic	410 ... ( 20.0 % )
Line Process Array Data ... Harmonic	512 ... ( 25.0 % )
Line Process Array Data ... Harmonic	615 ... ( 30.0 % )
Line Process Array Data ... Harmonic	717 ... ( 35.0 % )
Line Process Array Data ... Harmonic	820 ... ( 40.0 % )
Line Process Array Data ... Harmonic	922 ... ( 45.0 % )
Line Process Array Data ... Harmonic	1024 ... ( 50.0 % )
Line Process Array Data ... Harmonic	1127 ... ( 55.0 % )
Line Process Array Data ... Harmonic	1229 ... ( 60.0 % )
Line Process Array Data ... Harmonic	1332 ... ( 65.0 % )
Line Process Array Data ... Harmonic	1434 ... ( 70.0 % )

```

Line Process Array Data ... Harmonic    1536 ... ( 75.0 % )
Line Process Array Data ... Harmonic    1639 ... ( 80.0 % )
Line Process Array Data ... Harmonic    1741 ... ( 85.0 % )
Line Process Array Data ... Harmonic    1844 ... ( 90.0 % )
Line Process Array Data ... Harmonic    1946 ... ( 95.0 % )
Line Process Array Data ... Harmonic    2048 ... ( 99.9 % )

```

```

Simulation 1    Turning Band Line 1
Simulation 1    Turning Band Line 2
Simulation 1    Turning Band Line 3
Simulation 1    Turning Band Line 4
Simulation 1    Turning Band Line 5
Simulation 1    Turning Band Line 6
Simulation 1    Turning Band Line 7
Simulation 1    Turning Band Line 8
Simulation 1    Turning Band Line 9
Simulation 1    Turning Band Line 10
Simulation 1    Turning Band Line 11
Simulation 1    Turning Band Line 12
Simulation 1    Turning Band Line 13
Simulation 1    Turning Band Line 14
Simulation 1    Turning Band Line 15
Simulation 1    Turning Band Line 16

```

```

Output Filename = GAUSS1.DAT
The Sample Mean = -0.19667E+00
Sample Variance = 0.10864E+01

```

## Discussion of Results

The input files and plots of the output fields for the four stationary covariance models are shown on the following pages. The most obvious result of these simulations is the marked differences in the appearance of these random fields. Note that the fields generated with the exponential and Telis covariance functions are extremely “busy” compared to the Bessel (reasonably smooth transitions among neighboring values) and the Gaussian (very smooth) random fields. These differences can be explained via the integral scale analysis (section 2.1.1) or by examining the spectral distribution functions of these covariance models (Figure 4). The spectra of the exponential and Telis covariance models contain a great deal more high frequency information than the Bessel and Gaussian covariance models. In other words, when the spectrum encompasses high frequencies, there will be rapid spatial fluctuations over short distances as evidenced by these plots. The Gaussian model, which has its entire spectrum contained at frequencies considerably less than the other models, shows correspondingly fewer short range fluctuations and appears much smoother. This concept is illustrated further in section 4.3.3.

Figure 12 shows the theoretical and discrete variograms calculated for these random fields; although there is some scatter about the theoretical lines due to the limited sample size, these results show good agreement, indicating that TUBA preserves the desired correlation structure. The sample mean and variance statistics however, deviate markedly from their target values; the reason for this is explained in section 4.3.1.

Input Parameters For Gaussian Covariance Model [GAUSS1.INP]

```

      2          1=(x,y) Locations, 2=Gridded output
      2          1=Point Centered, 2=Block Centered
100.0000 100.0000 Maximum X and Y Field Dimensions
      100      100 Number of Nodes-X and Nodes-Y
NONE      Mask Filename
      1          1=Normal, 2=exp(X), 3=10**(X)
      2          0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC
      0.0000   1.0000 Desired Mean and Variance
      20.0000 20.0000 X and Y Direction Correlation Lengths
      1          1=Default TBM Parameters, 2=Enter Manually
GAUSS1.DAT Output Data Filename
      1          1=Unformatted, 2=Formatted Output
      2          1=Single Write Statement, 2=Line at a Time
      2          1=Marsaglia URNG, 2=Machine Indep URNG
      76651594 Seed for Random Number Generator
      1          Number of Realizations to be Simulated

! Field origin relative to TBM origin =      0.0000      0.0000
! Number of Turning Band Lines Equals =      16
! The Maximum Turning Band Line Length =     141.4214
! Turning Band Line Discretization Lgth =      1.0000
! Maximum Frequency for the Spectrum =     125.6637
! Number of Harmonics for the Spectrum =     2048
! Frequency Spacing in Spectral Domain =      0.0614
! Spatial Discretizations, DELX & DELY =      1.0000      1.0000
! No Pnts/correlation Length in X,Y Dir =     20.0      20.0
! Approx Number of Independent Samples =      6.2
!          The Sample Mean =     -0.1967
!          Sample Variance =      1.0864

```

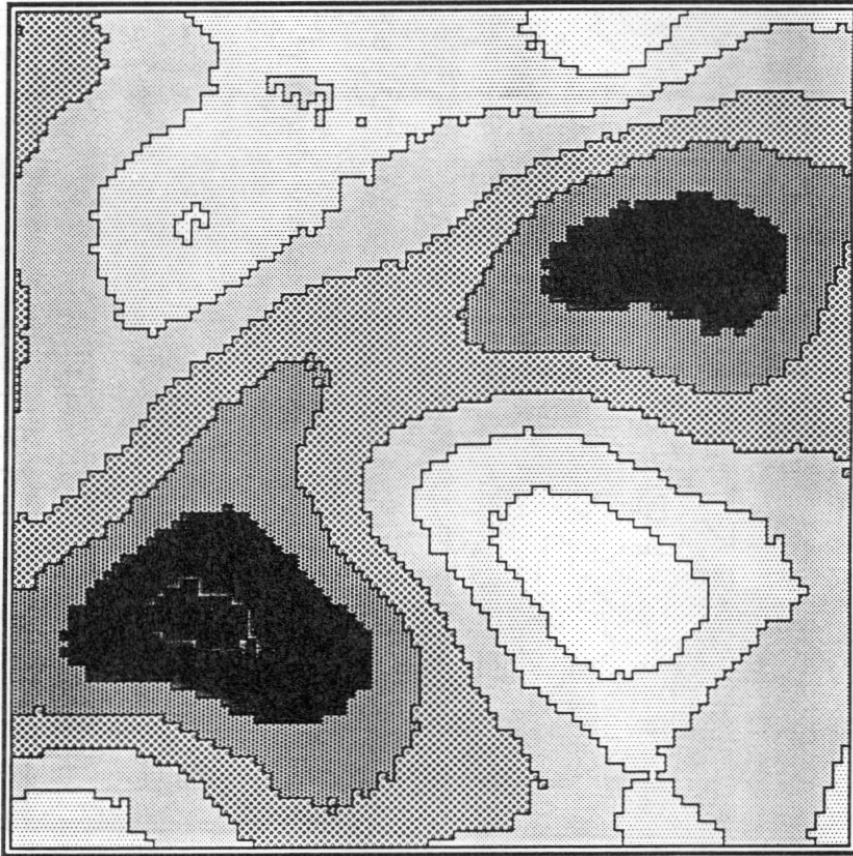


Figure 8. Random field generated using a Gaussian covariance model.

Input Parameters For Bessel Covariance Model [BESL1.INP]

```

      2          1=(x,y) Locations, 2=Gridded output
      2          1=Point Centered, 2=Block Centered
100.0000 100.0000 Maximum X and Y Field Dimensions
      100      100 Number of Nodes-X and Nodes-Y
NONE      Mask Filename
      1          1=Normal, 2=exp(X), 3=10**(X)
      3          0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC
.0000     1.0000 Desired Mean and Variance
12.0000   12.0000 X and Y Direction Correlation Lengths ← λ = λc / 1.65
      1          1=Default TBM Parameters, 2=Enter Manually
BESL1.DAT Output Data Filename
      1          1=Unformatted, 2=Formatted Output
      2          1=Single Write Statement, 2=Line at a Time
      2          1=Marsaglia URNG, 2=Machine Indep URNG
57632319 Seed for Random Number Generator
      1          Number of Realizations to be Simulated

! Field origin relative to TBM origin = .0000 .0000
! Number of Turning Band Lines Equals = 16
! The Maximum Turning Band Line Length = 141.4214
! Turning Band Line Discretization Lgth = .7500
! Maximum Frequency for the Spectrum = 100.5310
! Number of Harmonics for the Spectrum = 4096
! Frequency Spacing in Spectral Domain = .0245
! Spatial Discretizations, DELX & DELY = 1.0 1.0
! No Pnts/correlation Length in X,Y Dir = 12.0 12.0
! Approx Number of Independent Samples = 17.4
! The Sample Mean = -.9622
! Sample Variance = .7489

```

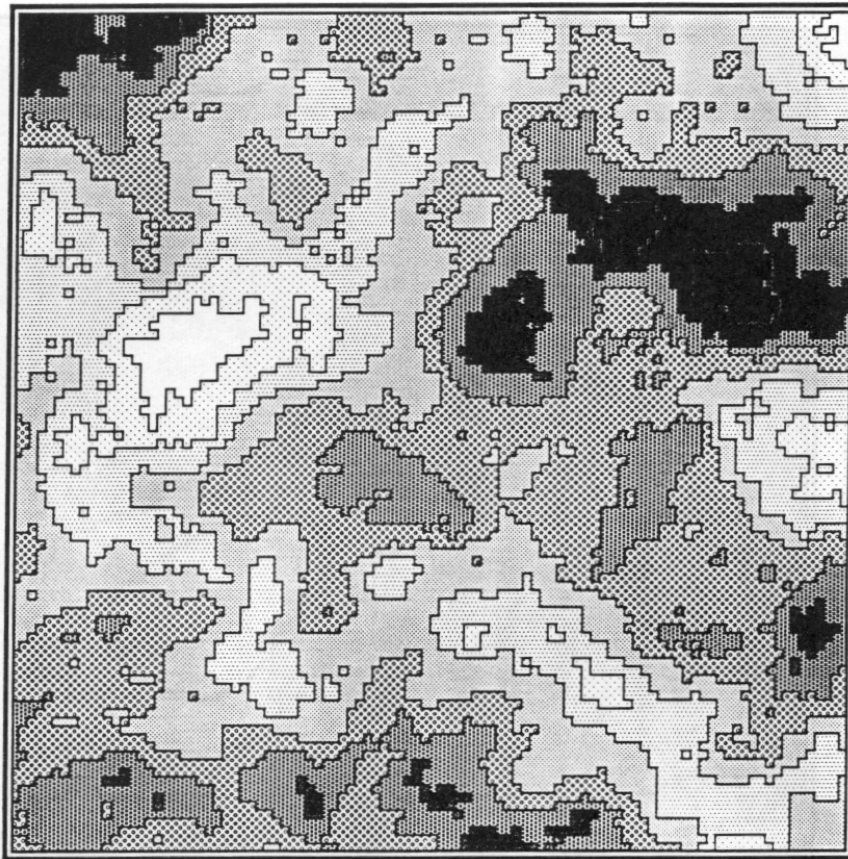


Figure 9. Random field generated using an Bessel covariance model.

Input Parameters For Exponential Covariance Model [EXP01.INP]

```

      2          1=(x,y) Locations, 2=Gridded output
      2          1=Point Centered, 2=Block Centered
100.0000 100.0000 Maximum X and Y Field Dimensions
      100      100 Number of Nodes-X and Nodes-Y
NONE      Mask Filename
      1          1=Normal, 2=exp(X), 3=10**(X)
      1          0=User, 1=Exp, 2=Gauss, 3=Besl, 4=Telis, 5=GC
0.0000    1.0000 Desired Mean and Variance
20.0000   20.0000 X and Y Direction Correlation Lengths
      1          1=Point Process, 2=Areal Average Process ←see sec 4.2.4
      1          1=Default TBM Parameters, 2=Enter Manually
EXP01.DAT Output Data Filename
      1          1=Unformatted, 2=Formatted Output
      1          1=Single Write Statement, 2=Line at a Time
      2          1=Marsaglia URNG, 2=Machine Indep URNG
57756341 Seed for Random Number Generator
      1          Number of Realizations to be Simulated

! Field origin relative to TBM origin =      .0000      .0000
! Number of Turning Band Lines Equals =      16
! The Maximum Turning Band Line Length =     141.4214
! Turning Band Line Discretization Lgth =      1.0000
! Maximum Frequency for the Spectrum =     125.6637
! Number of Harmonics for the Spectrum =     4096
! Frequency Spacing in Spectral Domain =      .0307
! Spatial Discretizations, DELX & DELY =      1.0      1.0
! No Pnts/correlation Length in X,Y Dir =     20.0      20.0
! Approx Number of Independent Samples =      6.3
!                               The Sample Mean =     -.1282
!                               Sample Variance =      .7995

```

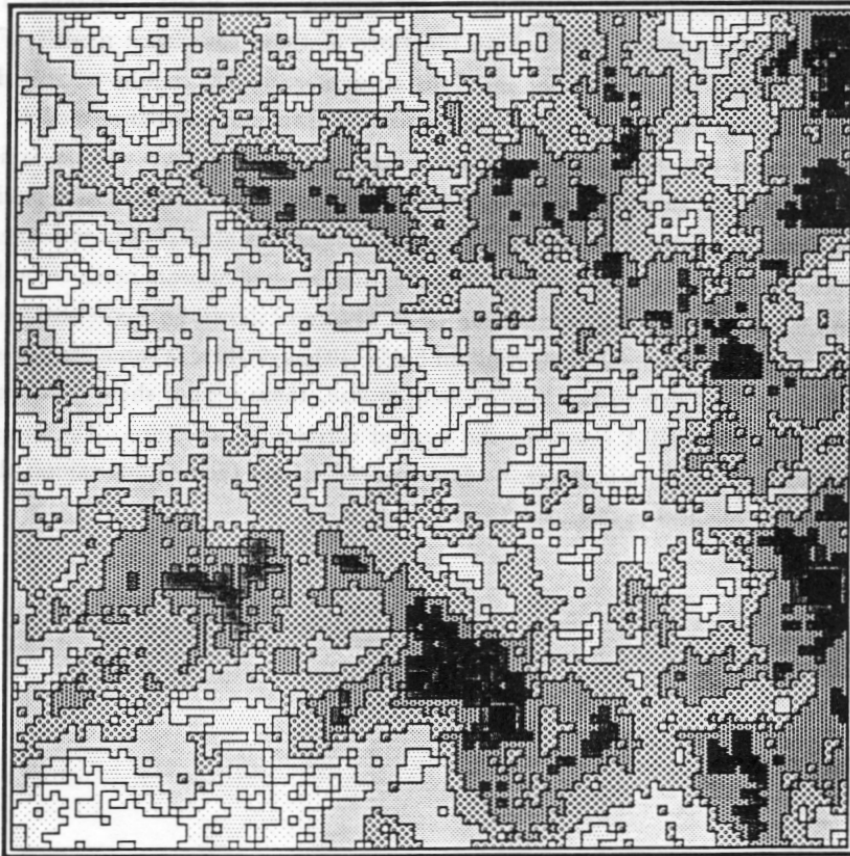


Figure 10. Random field generated using an exponential covariance model.

Input Parameters For Telis Covariance Model [ TELIS1.INP ]

```

      2          1=(x,y) Locations, 2=Gridded output
      2          1=Point Centered, 2=Block Centered
100.0000 100.0000 Maximum X and Y Field Dimensions
      100      100 Number of Nodes-X and Nodes-Y
NONE      Mask Filename
      1          1=Normal, 2=exp(X), 3=10**(X)
      4          0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC
.0000     1.0000 Desired Mean and Variance
27.0000   27.0000 X and Y Direction Correlation Lengths ← λ =  $\frac{\lambda_c}{0.75}$ 
      1          1=Default TBM Parameters, 2=Enter Manually
TELIS1.DAT Output Data Filename
      1          1=Unformatted, 2=Formatted Output
      2          1=Single Write Statement, 2=Line at a Time
      2          1=Marsaglia URNG, 2=Machine Indep URNG
76548921   Seed for Random Number Generator
      1          Number of Realizations to be Simulated

! Field origin relative to TBM origin =      .0000      .0000
! Number of Turning Band Lines Equals =      16
! The Maximum Turning Band Line Length =     141.4214
! Turning Band Line Discretization Lgth =      1.0000
! Discretization Dstnce for MA Process =      1.0000
! Number of Output Pnts Along the Line =      142
! Spatial Discretizations, DELX & DELY =      1.0      1.0
! No Pnts/correlation Length in X,Y Dir =     27.0      27.0
! Approx Number of Independent Samples =      3.4
!                               The Sample Mean =     -0.3297
!                               Sample Variance =      0.6816

```

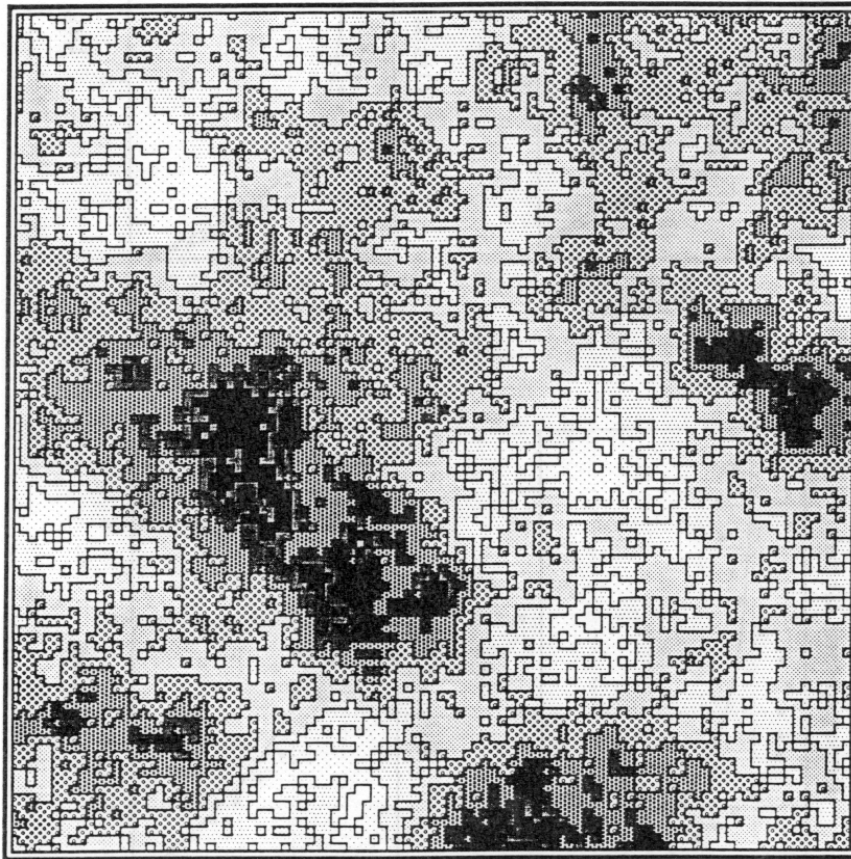


Figure 11. Random field generated using a Telis covariance model.

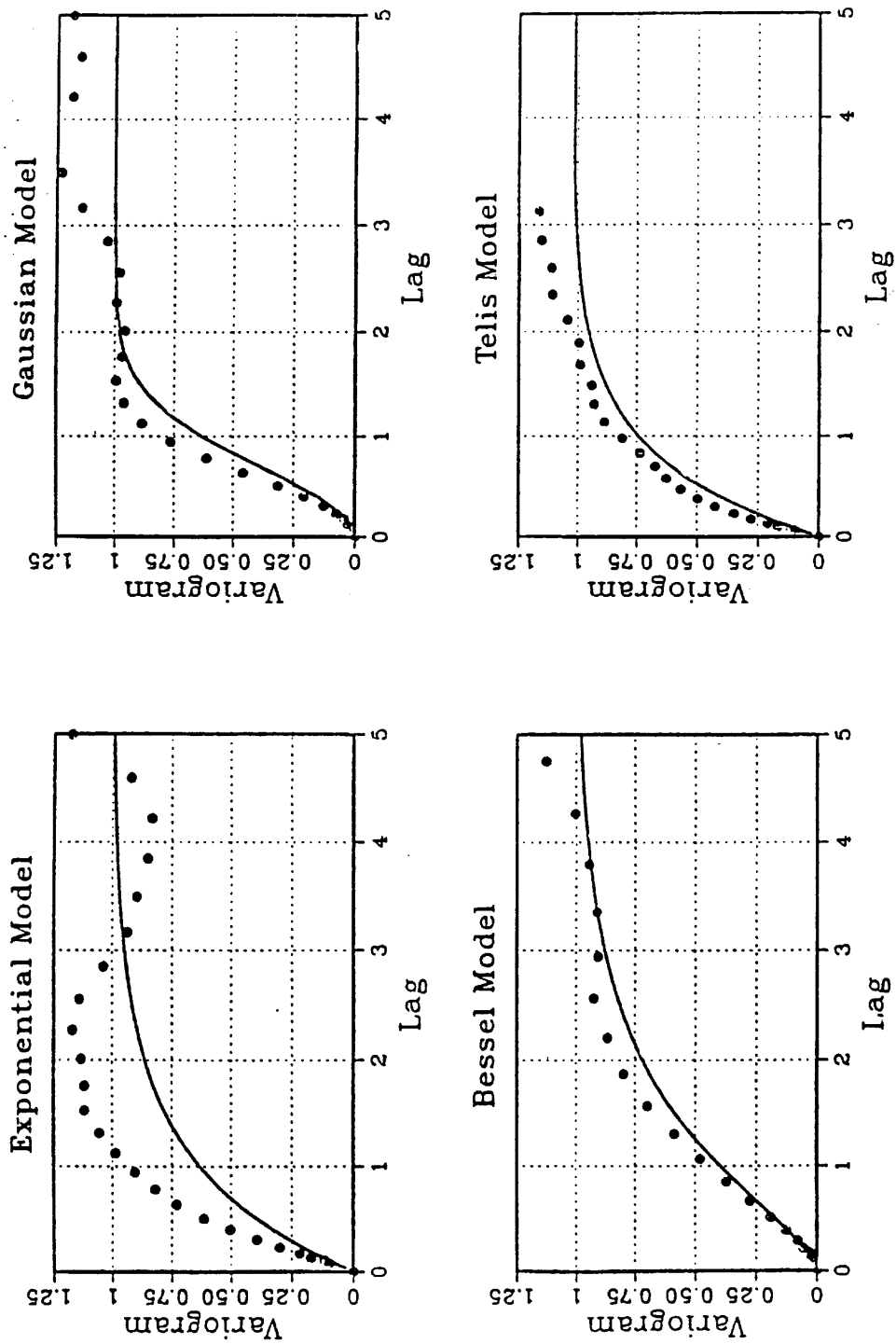


Figure 12. Variogram estimates for TUBA generated stationary random fields. Solid lines are the theoretical variogram functions.



### § 4.2.2 Generation of Anisotropic Random Fields

It was noted in section 2.7 that anisotropic random fields of the ellipsoidal type are generated via coordinate transformations. The input control stream for generating a Gaussian random field having an  $x:y = 5:1$  anisotropic correlation structure is shown above Figure 14. All that is required is to specify the  $x$  and  $y$  direction correlation lengths proportional to the anisotropy.

Directional variogram estimates of the output field are shown in Figure 13. The Variogram plots show that the sill is reached at approximately  $(.8)(60) = 48$  units in the horizontal direction and  $(.15)(60) = 9$  units in the vertical direction; this analysis reflects the desired 5:1 ratio for the horizontal to vertical anisotropy. The field is plotted in Figure 14 and clearly shows its anisotropic character.

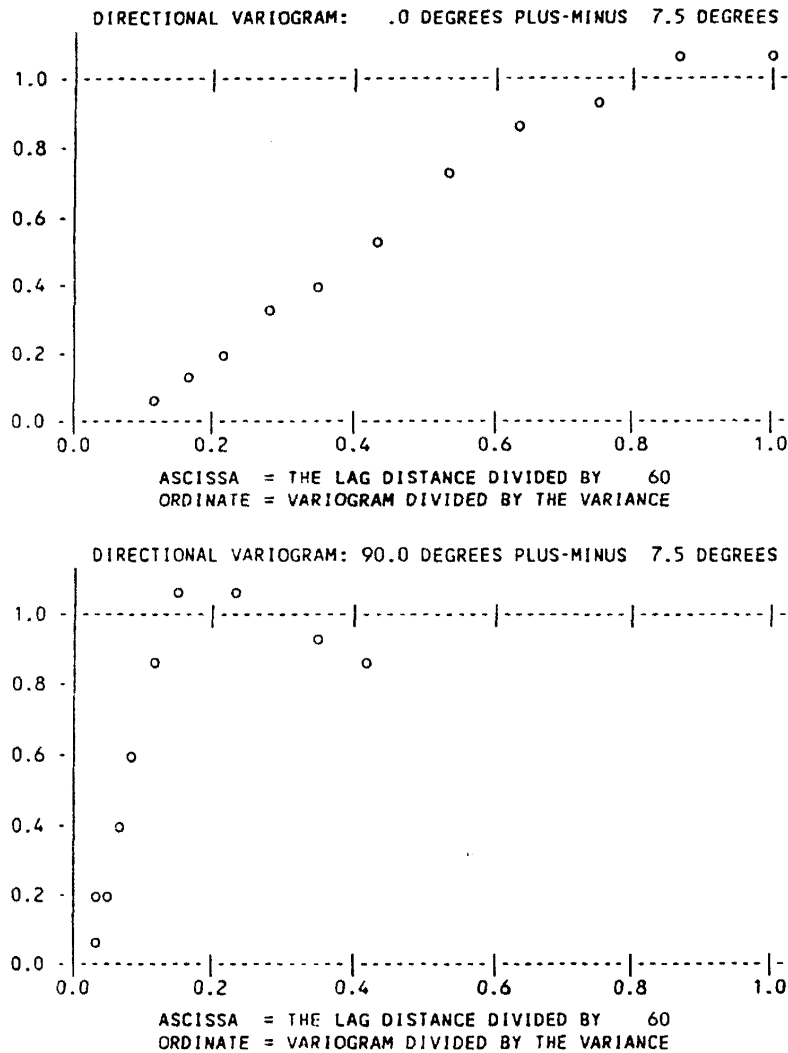


Figure 13. Directional variogram estimates for anisotropic Gaussian field.

Input Parameters For Anisotropic Gaussian Field [GAUSS2.INP]

```

      2          1=(x,y) Locations, 2=Gridded output
      2          1=Point Centered, 2=Block Centered
100.0000 100.0000 Maximum X and Y Field Dimensions
      100      100 Number of Nodes-X and Nodes-Y
NONE          Mask Filename
      1          1=Normal, 2=exp(X), 3=10**(X)
      2          0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC
.0000      1.0000 Desired Mean and Variance
35.0000      7.0000 X and Y Direction Correlation Lengths ← anisotropy
      1          1=Default TBM Parameters, 2=Enter Manually
GAUSS2.DAT   Output Data Filename
      1          1=Unformatted, 2=Formatted Output
      2          1=Single Write Statement, 2=Line at a Time
      2          1=Marsaglia URNG, 2=Machine Indep URNG
84756542     Seed for Random Number Generator
      1          Number of Realizations to be Simulated

! Field origin relative to TBM origin =      .0000      .0000
! Number of Turning Band Lines Equals =      16
! The Maximum Turning Band Line Length =      141.4214
! Turning Band Line Discretization Lgth =      .2000
! Maximum Frequency for the Spectrum =      219.9115
! Number of Harmonics for the Spectrum =      4096
! Frequency Spacing in Spectral Domain =      .0537
! Spatial Discretizations, DELX & DELY =      1.0      1.0
! No Pnts/correlation Length in X,Y Dir =      35.0      7.0
! Approx Number of Independent Samples =      10.2
!          The Sample Mean =      .5460
!          Sample Variance =      .7596

```

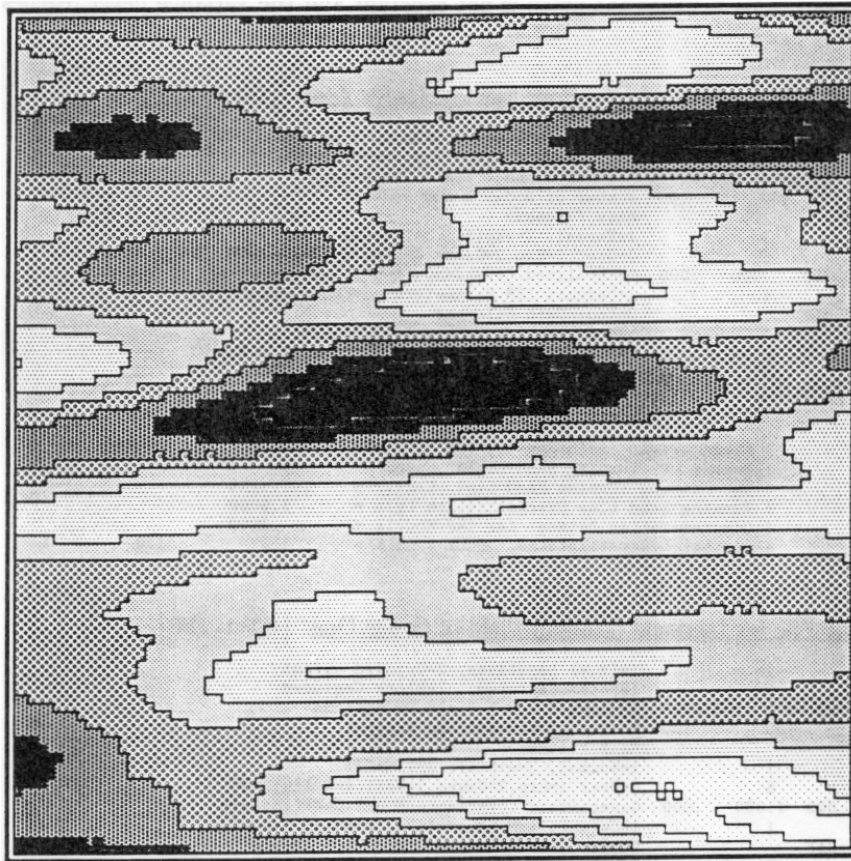


Figure 14. Anisotropic field generated using a Gaussian covariance model.

### § 4.2.3 Generation of Non-stationary Random Fields

In this section we generate three non-stationary random fields: an IRF-0, an IRF-1, and an IRF-2 (see section 2.5). The fields are generated onto a block-centered finite difference grid as before and the results are plotted in Figures 15 through 17. The input files used to generate these fields are shown just above those figures. Note that the same random number generator seed is used for all three fields; only the order of the intrinsic random field (as determined by the Generalized Covariance model coefficients) changes from one simulation to the next. Also, note that the mean and variance statistics are not calculated for these random fields because the mean is not constant and the variance never reaches a sill.

#### Discussion of Results

Notice that the spatial variability patterns in the intrinsic random fields depicted in Figures 15 through 17 progress from more noisy to less noisy as the order of the IRF increases. This is because the higher-order fields filter out higher-order polynomial trends (see section 2.5 and Table 2).

Each of these fields were analysed using AKRIP (*Kafritsas and Bras, [1984]*); AKRIP is a computer code for spatial structure analysis and kriging in two-dimensions using Intrinsic Random Field theory. Approximately 100 sample points were taken randomly from each output file and used as input to AKRIP. Both the order of the intrinsic random field and the Generalized Covariance model coefficients were estimated by AKRIP and found to be in agreement with the parameters used to generate the fields with TUBA.

Input File for Intrinsic Random Field of Order Zero [GC-0.INP]

```

          2          1=(x,y) Locations, 2=Gridded output
          2          1=Point Centered, 2=Block Centered
100.0000 100.0000 Maximum X and Y Field Dimensions
          100          100 Number of Nodes-X and Nodes-Y
NONE      Mask Filename
          1          1=Normal, 2=exp(X), 3=10**(X)
          5          0=User, 1=Exp, 2=Gauss, 3=Bes1, 4=Telis, 5=GC
1.000    .000    .000 Generalized Covariance Model Coefficients
          1          1=Default TBM Parameters, 2=Enter Manually
GC-0.DAT Output Data Filename
          1          1=Unformatted, 2=Formatted Output
          2          1=Single Write Statement, 2=Line at a Time
          2          1=Marsaglia URNG, 2=Machine Indep URNG
28104199 Seed for Random Number Generator
          1          Number of Realizations to be Simulated

! Field origin relative to TBM origin =      .0000      .0000
! Number of Turning Band Lines Equals =      16
! The Maximum Turning Band Line Length =    141.4214
! Turning Band Line Discretization Lgth =     1.0000
! Discretization Dstnce for IRF Models =     .2500
! Spatial Discretizations, DELX & DELY =     1.0      1.0

```

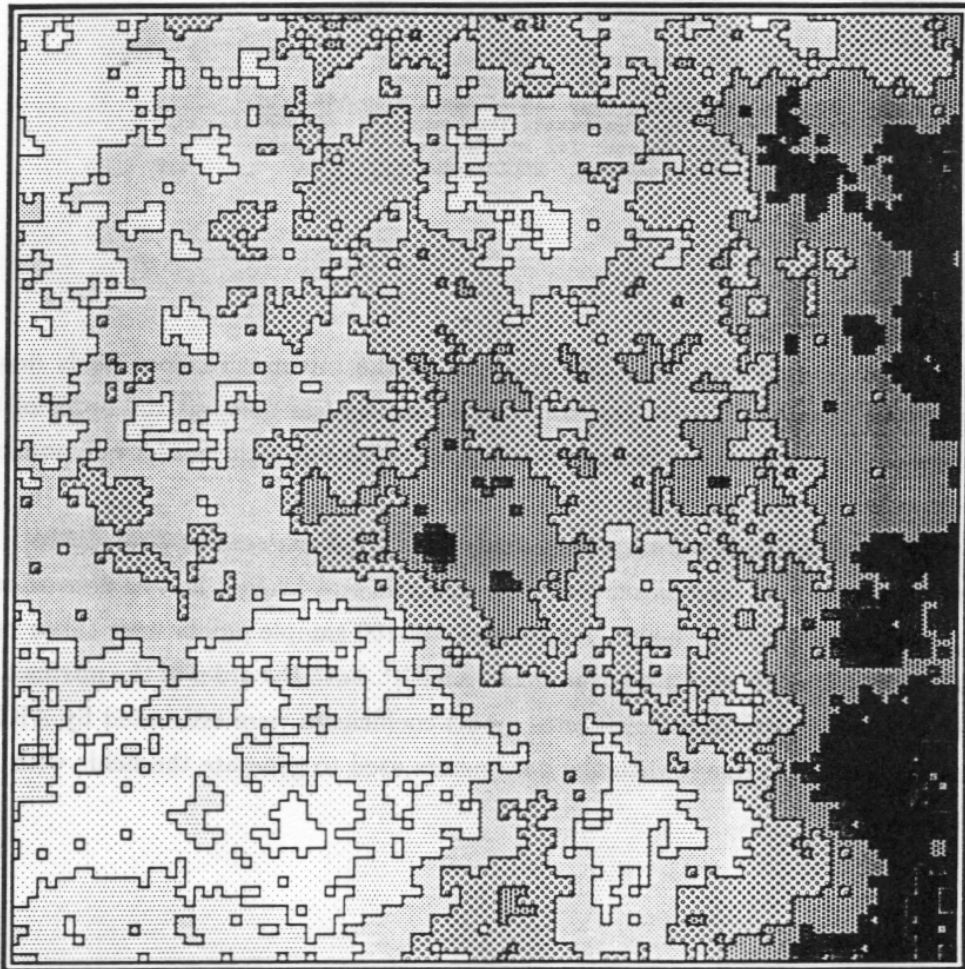


Figure 15. Random field generated using a Generalized Covariance model for an Intrinsic Random Field of Order 0 (IRF-0).

Input File for Intrinsic Random Field of Order One [GC-1.INP]

```

          2          1=(x,y) Locations, 2=Gridded output
          2          1=Point Centered, 2=Block Centered
    100.0000  100.0000 Maximum X and Y Field Dimensions
          100          100 Number of Nodes-X and Nodes-Y
NONE      NONE      Mask Filename
          1          1=Normal, 2=exp(X), 3=10**(X)
          5          0=User, 1=Exp, 2=Gauss, 3=Besl, 4=Telis, 5=GC
    .000  1.000  .000 Generalized Covariance Model Coefficients
          1          1=Default TBM Parameters, 2=Enter Manually
GC-1.DAT  GC-1.DAT  Output Data Filename
          1          1=Unformatted, 2=Formatted Output
          2          1=Single Write Statement, 2=Line at a Time
          2          1=Marsaglia URNG, 2=Machine Indep URNG
    28104199 Seed for Random Number Generator
          1          Number of Realizations to be Simulated

! Field origin relative to TBM origin =      .0000      .0000
! Number of Turning Band Lines Equals =      16
! The Maximum Turning Band Line Length =    141.4214
! Turning Band Line Discretization Lgth =    1.0000
! Discretization Dstnce for IRF Models =    .2500
! Spatial Discretizations, DELX & DELY =    1.0      1.0

```

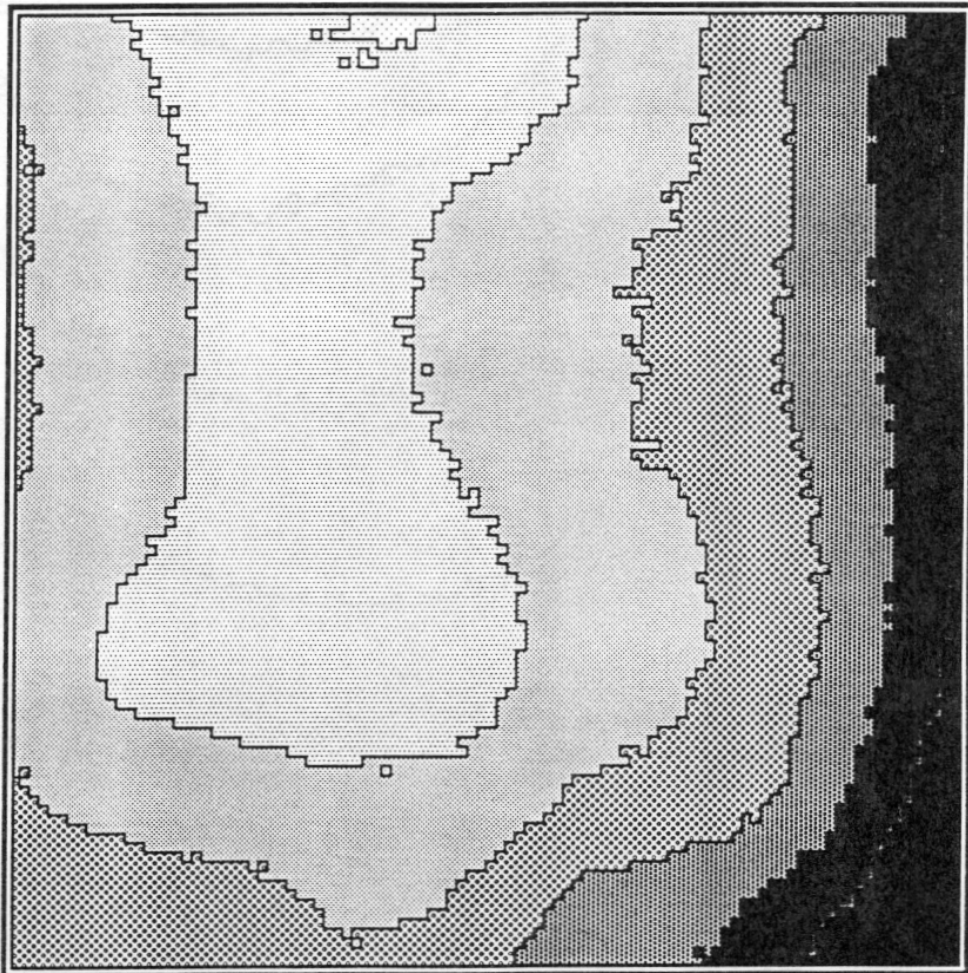


Figure 16. Random field generated using a Generalized Covariance model for an Intrinsic Random Field of Order 1 (IRF-1).

Input File for Intrinsic Random Field of Order Two [GC-2.INP]

```

          2          1=(x,y) Locations, 2=Gridded output
          2          1=Point Centered, 2=Block Centered
100.0000 100.0000 Maximum X and Y Field Dimensions
          100          100 Number of Nodes-X and Nodes-Y
NONE      Mask Filename
          1          1=Normal, 2=exp(X), 3=10**(X)
          5          0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC
.000      .000  1.000 Generalized Covariance Model Coefficients
          1          1=Default TBM Parameters, 2=Enter Manually
GC-2.DAT  Output Data Filename
          1          1=Unformatted, 2=Formatted Output
          2          1=Single Write Statement, 2=Line at a Time
          2          1=Marsaglia URNG, 2=Machine Indep URNG
28104199  Seed for Random Number Generator
          1          Number of Realizations to be Simulated

! Field origin relative to TBM origin =      .0000      .0000
! Number of Turning Band Lines Equals =      16
! The Maximum Turning Band Line Length =    141.4214
! Turning Band Line Discretization Lgth =     1.0000
! Discretization Dstnce for IRF Models =     .2500
! Spatial Discretizations, DELX & DELY =     1.0      1.0

```

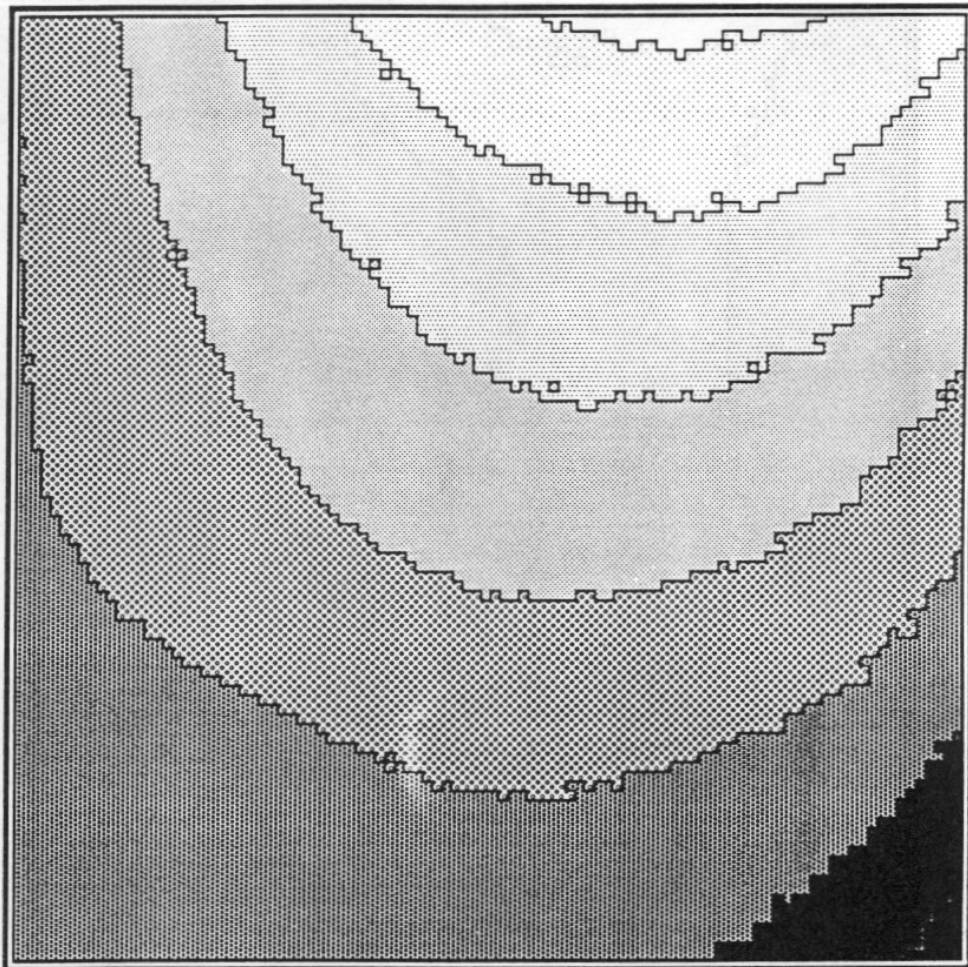


Figure 17. Random field generated using a Generalized Covariance model for an Intrinsic Random Field of Order 2 (IRF-2).

#### § 4.2.4 Generation of Areal Average Random Fields

In this section we generate areal average random fields for different size rectangular averaging areas (see section 2.6) to illustrate the smoothing effect that averaging has. This option is available only for the exponential covariance model (although it could be programmed in for the other models as well) and we use for comparison, the exponential field generated in section 4.2.1 (Figure 10). Two areal average random fields are generated here; the only difference between the input control streams of these files and the one in section 4.2.1 is the size of the averaging area: The field generated in section 4.2.1 was specified as a “point process” (i.e., no averaging) whereas these fields use areal averaging where the averaging area is a square region whose sides are of length  $0.25\lambda$  and  $1.0\lambda$  where  $\lambda$  is the correlation length of the process. The input for these fields are shown above the resulting fields which are plotted in Figures 18 and 19.

##### Discussion of Results

Figures 18 and 19 illustrate the smoothing effect of larger and larger averaging areas; these should be compared with Figure 10, the same field with no averaging. It should be understood that these figures illustrate *moving areal averages* where the averaging areas are permitted to overlap each other as the rectangular averaging window slides along from point to point in the random field.

In practical applications, one may want to use areal averaging as a means of representing material properties which remain uniform within the blocks of a finite difference grid; in this case no overlapping of the averaging areas would occur. To illustrate the spatial variability patterns this type of system would exhibit, we generate the same field again using averaging rectangles of size  $0.25\lambda$ , but generate at fewer locations (a coarser grid) so that the averaging rectangles do not overlap. The input file for this case<sup>†</sup> is shown just above the plotted result, Figure 20.

---

<sup>†</sup>Manual entry of the Turning Band parameters was required for this case in order to match the parameter values shown at the bottom of the `EXP02.INP` and `EXP03.INP` list files. This is because only 2048 harmonics would have been required to generate the field onto the smaller 20 by 20 grid; had the default Turning Band parameters been used, a field would have been generated having the proper statistics, but the spatial variability patterns would not correspond to the other examples in this section because they required 4096 harmonics for the line process.

What is not apparent from this example is that the FFT line generation method was used even though the Turning Band parameters were entered manually. This is because TUBA checks to see if the user specified number of harmonics =  $2^n$  for some  $n$ ; if it is, TUBA uses the much faster FFT method rather than the method of Shinozuka and Jan. If the number of harmonics were input as 4095 or 4097, essentially the same results would be obtained but at much greater computational expense.



Input Data for Areal Average Field [EXP02.INP]

```

2          1=(x,y) Locations, 2=Gridded output
2          1=Point Centered, 2=Block Centered
100.0000  100.0000 Maximum X and Y Field Dimensions
100       100   Number of Nodes-X and Nodes-Y
NONE      Mask Filename
1         1=Normal, 2=exp(X), 3=10**(X)
1         0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC
.0000    1.0000 Desired Mean and Variance
20.0000  20.0000 X and Y Direction Correlation Lengths
2         1=Point Process, 2=Areal Average Process
5.0000   5.0000 X and Y Dimensions of Averaging Area ← = 0.25λ
1         1=Default TBM Parameters, 2=Enter Manually
EXP02.DAT Output Data Filename
1         1=Unformatted, 2=Formatted Output
2         1=Single Write Statement, 2=Line at a Time
2         1=Marsaglia URNG, 2=Machine Indep URNG
57756341 Seed for Random Number Generator
1         Number of Realizations to be Simulated

! Field origin relative to TBM origin =      .0000      .0000
! Number of Turning Band Lines Equals =      16
! The Maximum Turning Band Line Length =    141.4214
! Turning Band Line Discretization Lgth =    1.0000
! Maximum Frequency for the Spectrum =      125.6637
! Number of Harmonics for the Spectrum =     4096
! Frequency Spacing in Spectral Domain =     .0307
! Spatial Discretizations, DELX & DELY =     1.0      1.0
! No Pnts/correlation Length in X,Y Dir =    20.0     20.0
! Approx Number of Independent Samples =     6.3
! The Sample Mean =                          -.1276
! Sample Variance =                          .6692

```

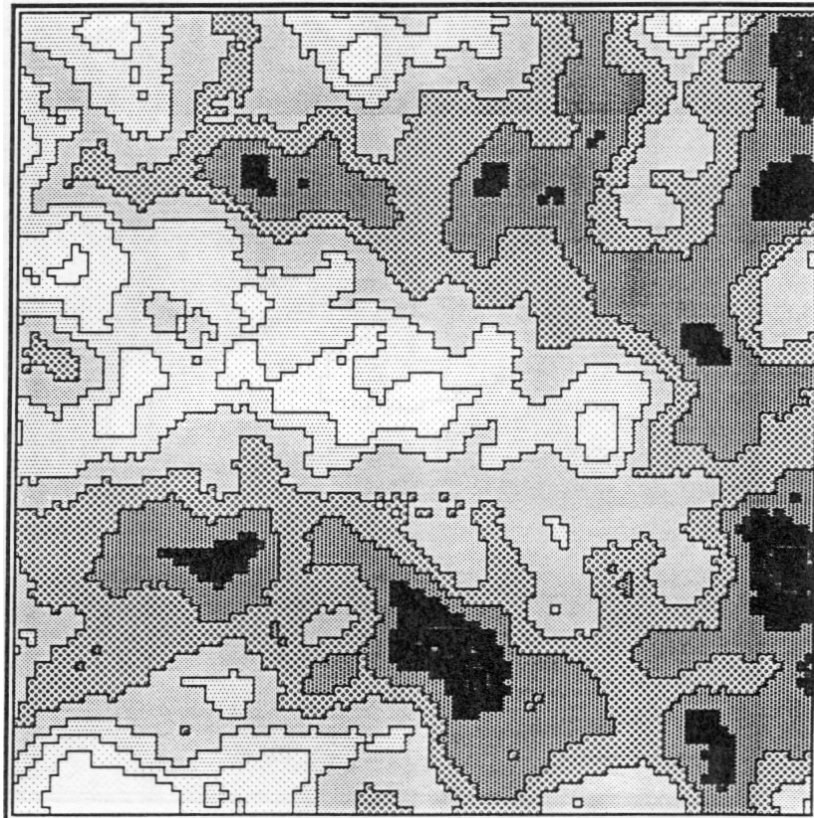


Figure 18. Areal averaged exponential random field with averaging window equal to 1/4th the correlation length.



Input Data for Areal Average Field [EXP03.INP]

```

2          1=(x,y) Locations, 2=Gridded output
2          1=Point Centered, 2=Block Centered
100.0000  100.0000 Maximum X and Y Field Dimensions
100       100   Number of Nodes-X and Nodes-Y
NONE      Mask Filename
1         1=Normal, 2=exp(X), 3=10**(X)
1         0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC
.0000    1.0000 Desired Mean and Variance
20.0000  20.0000 X and Y Direction Correlation Lengths
2        1=Point Process, 2=Areal Average Process
20.0000  20.0000 X and Y Dimensions of Averaging Area ← = 1.0λ
1        1=Default TBM Parameters, 2=Enter Manually
EXP03.DAT Output Data Filename
1        1=Unformatted, 2=Formatted Output
2        1=Single Write Statement, 2=Line at a Time
2        1=Marsaglia URNG, 2=Machine Indep URNG
57756341 Seed for Random Number Generator
1        Number of Realizations to be Simulated

! Field origin relative to TBM origin = .0000 .0000
! Number of Turning Band Lines Equals = 16
! The Maximum Turning Band Line Length = 141.4214
! Turning Band Line Discretization Lgth = 1.0000
! Maximum Frequency for the Spectrum = 125.6637
! Number of Harmonics for the Spectrum = 4096
! Frequency Spacing in Spectral Domain = .0307
! Spatial Discretizations, DELX & DELY = 1.0 1.0
! No Pnts/correlation Length in X,Y Dir = 20.0 20.0
! Approx Number of Independent Samples = 6.3
! The Sample Mean = -.1144
! Sample Variance = .3494

```

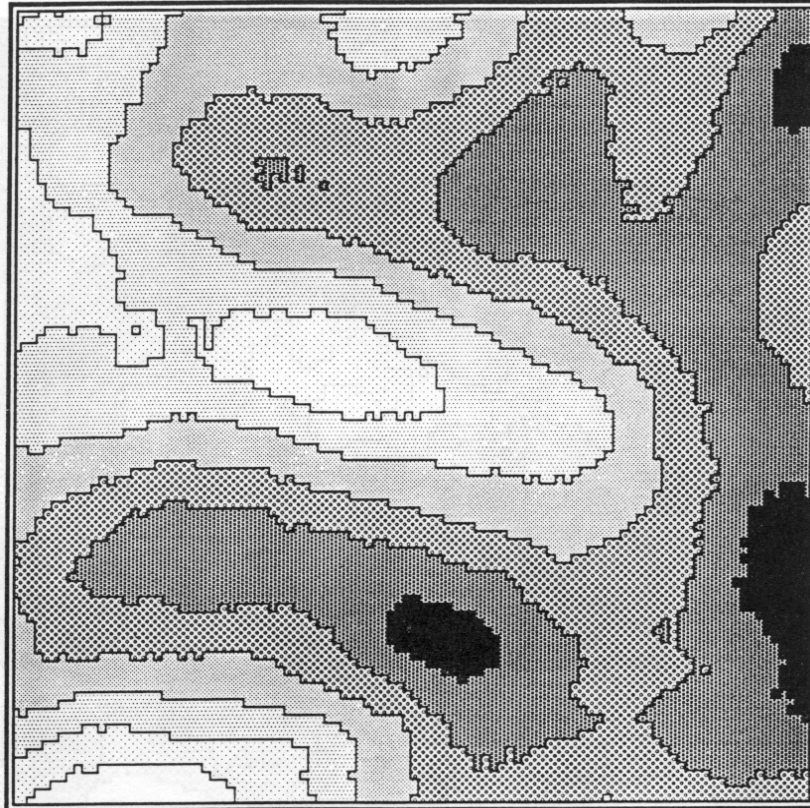


Figure 19. Areal averaged exponential random field with averaging window equal to the correlation length.

Input Data for Non-overlapping Areal Average Process [EXP04.INP]

```

2          1=(x,y) Locations, 2=Gridded output
2          1=Point Centered, 2=Block Centered
100.0000  100.0000 Maximum X and Y Field Dimensions
20         20         Number of Nodes-X and Nodes-Y          ← note grid size
NONE      Mask Filename
1          1=Normal, 2=exp(X), 3=10**(X)
1          0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC
.0000     1.0000     Desired Mean and Variance
20.0000   20.0000   X and Y Direction Correlation Lengths
2          1=Point Process, 2=Areal Average Process
5.0000    5.0000    X and Y Dimensions of Averaging Area
2          1=Default TBM Parameters, 2=Enter Manually      ← note
16         Number of Turning Band Lines
1.0000    TBM Line Discretization Distance                ← key
4096      Nbr of Harmonics for Discretizing Spectrum       ← key
.0000     .0000     Field Origin Relative to TBM Origin
141.4214  Maximum Turning Band Line Length
EXP04.DAT Output Data Filename
1          1=Unformatted, 2=Formatted Output
2          1=Single Write Statement, 2=Line at a Time
2          1=Marsaglia URNG, 2=Machine Indep URNG
57756341  Seed for Random Number Generator
1          Number of Realizations to be Simulated

! Field origin relative to TBM origin = .0000 .0000
! Number of Turning Band Lines Equals = 16
! The Maximum Turning Band Line Length = 141.4214
! Turning Band Line Discretization Lgth = 1.0000
! Maximum Frequency for the Spectrum = 125.6637
! Number of Harmonics for the Spectrum = 4096
! Frequency Spacing in Spectral Domain = .0307
! Spatial Discretizations, DELX & DELY = 5.0 5.0
! No Pnts/correlation Length in X,Y Dir = 4.0 4.0
! Approx Number of Independent Samples = 6.3
! The Sample Mean = -.1222
! Sample Variance = .6772

```

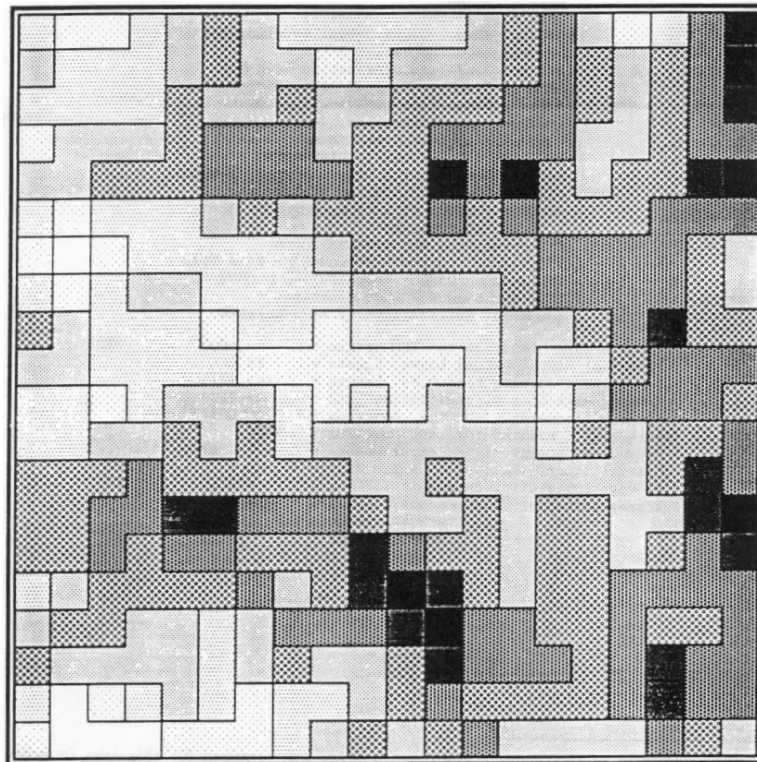


Figure 20. Areal averaged exponential random field with non-overlapping averaging windows equal to  $\frac{1}{4}$ th the correlation length. Compare this result with Figure 18.

## § 4.2.5 Generating at Arbitrary Locations in Space

In this example, we generate a random field on a small (11 x 9) point centered finite difference grid and on a finite element grid composed of quadrilateral elements whose nodes are coincident with some of the nodes of the finite difference grid (see Figure 21). In order to obtain identical field values between the nodes of the finite difference and finite element grids, the distance from the Turning Bands origin to the most remote point in the output field, TBMX, must be the same for both cases. This is because the internal parameter TBMX determines how many discrete values of the line process will be generated along each Turning Band line; since calls to the random number generator are made for each line process, the output fields will be different unless TBMX is the same for both cases.<sup>†</sup> This condition is established by asking TUBA to generate at an extra point which is not part of the finite element grid; that point is the upper right hand corner node (the most remote node) of the finite difference grid. Shown below is the input for generating the field values at the nodes of the finite difference grid, and the contents of FE-GRID.XYC, the input file containing the nodal coordinates of the finite element grid.

<u>Input data for small finite difference grid [ FD-GRID. INP ]</u>			<u>FD-GRID.XYC</u>
2		1=(x,y) Locations, 2=Gridded output	6,8
1		1=Point Centered, 2=Block Centered	8,7
10.0000	8.0000	Maximum X and Y Field Dimensions	9,6
11	9	Number of Nodes-X and Nodes-Y	10,4
NONE		Mask Filename	9,2
1		1=Normal, 2=exp(X), 3=10**(X)	5,8
4		0=User,1=Exp,2=Gauss,3=Bes1,4=Telis,5=GC	6,7
.0000	1.0000	Desired Mean and Variance	7,6
5.0000	5.0000	X and Y Direction Correlation Lengths	7,4
1		1=Default TBM Parameters, 2=Enter Manually	7,1
FD-GRID.DAT		Output Data Filename	4,8
2		1=Unformatted, 2=Formatted Output	5,7
(11F8.3)		Output Format for Writing Data to Disk	4,6
2		1=Single Write Statement, 2=Line at a Time	4,5
2		1=First Row to Last, 2=Last Row to First	5,5
2		1=Marsaglia URNG, 2=Machine Indep URNG	5,3
52379164		Seed for Random Number Generator	4,0
1		Number of Realizations to be Simulated	2,8
			3,7
!	Field origin relative to TBM origin =	.0000 .0000	3,6
!	Number of Turning Band Lines Equals =	16	3,5
!	The Maximum Turning Band Line Length =	12.8062	3,3
!	Turning Band Line Discretization Lgth =	.3125	1,1
!	Discretization Dstnce for MA Process =	.2500	1,7
!	Number of Output Pnts Along the Line =	41	1,5
!	Spatial Discretizations, DELX & DELY =	1.0 1.0	1,4
!	No Pnts/correlation Length in X,Y Dir =	5.0 5.0	0,3
!	Approx Number of Independent Samples =	.8	0,5
!	The Sample Mean =	1.4008	10,8
!	Sample Variance =	.9697	

<sup>†</sup> Be careful – this works for this case where the field is generated via a moving average process (i.e., the Telis model). More is involved when a spectral method is used for generation of the line processes (see footnote below the EXPO4. INP list file, section 4.2.4)

Listed below is the output data file for the finite difference grid. These gridded values are shown again in Figure 21 for clarity of presentation; the boxed nodal values correspond to the nodes of the finite element grid. The input parameters and output file for the finite element grid are also listed below. These results show that the simulations yield consistent output values when the fields are generated at identical points in space. This method of generating field values at specified locations could be used to obtain values *between* the node points if one is careful about how the simulation parameters are specified (see section 4.2.6).

Output Data For The Finite Difference Grid [FD-GRID.DAT]

3.303	2.966	1.772	2.508	2.091	2.435	1.556	.760	.127	.535	.130
3.046	3.132	1.807	2.838	2.046	2.811	2.623	1.471	.861	-.162	.423
2.239	2.201	3.421	2.967	2.597	3.109	2.693	1.527	.604	.888	.647
2.110	2.536	3.032	3.369	3.055	2.694	1.622	1.713	.820	1.612	1.356
.786	1.559	2.392	2.241	1.848	2.400	1.443	.867	.880	1.725	1.554
.198	.829	2.056	1.271	2.207	1.193	1.656	.166	.595	1.597	1.799
-.602	.198	.853	-.150	.329	1.513	1.259	.193	1.143	1.065	.738
.619	-.496	.265	.846	.431	1.167	.961	.999	1.206	.802	-.232
.555	.606	.358	.516	.450	1.418	1.417	1.361	1.066	.346	.328

Input Data For Finite Element Grid [FE-GRID.INP]

```

1          1=(x,y) Locations, 2=Gridded output
FE-GRID.XYC      Input Filename for (x,y) Locations
1          1=Normal, 2=exp(X), 3=10**(X)
4          0=User,1=Exp,2=Gauss,3=Bes1,4=Telis,5=GC
.0000      1.0000  Desired Mean and Variance
5.0000      5.0000  X and Y Direction Correlation Lengths
1          1=Default TBM Parameters, 2=Enter Manually
FE-GRID.XYZ      Output Data Filename
2          1=Output Only Z, 2=Output X,Y, and Z
2          1=Unformatted, 2=Formatted Output
(3(6X,2F6.1,F8.3))  Output Format for Writing Data to Disk      ← note format
2          1=Marsaglia URNG, 2=Machine Indep URNG
52379164      Seed for Random Number Generator
1          Number of Realizations to be Simulated

! Field origin relative to TBM origin =      .0000      .0000
! Number of Turning Band Lines Equals =      16
! The Maximum Turning Band Line Length =      12.8062
! Turning Band Line Discretization Lgth =      .3125
! Discretization Dstnce for MA Process =      .2500
! Number of Output Pnts Along the Line =      41
!          The Sample Mean =      1.7467
!          Sample Variance =      1.0551

```

Output Data For Finite Element Grid [FE-GRID.XYZ]

6.0	8.0	1.556	8.0	7.0	.861	9.0	6.0	.888
10.0	4.0	1.554	9.0	2.0	1.065	5.0	8.0	2.435
6.0	7.0	2.623	7.0	6.0	1.527	7.0	4.0	.867
7.0	1.0	.999	4.0	8.0	2.091	5.0	7.0	2.811
4.0	6.0	2.597	4.0	5.0	3.055	5.0	5.0	2.694
5.0	3.0	1.193	4.0	.0	.450	2.0	8.0	1.772
3.0	7.0	2.838	3.0	6.0	2.967	3.0	5.0	3.369
3.0	3.0	1.271	1.0	1.0	-.496	1.0	7.0	3.132
1.0	5.0	2.536	1.0	4.0	1.559	.0	3.0	.198
.0	5.0	2.110	10.0	8.0	.130			

Compare these numbers to the boxed values in Figure 21.

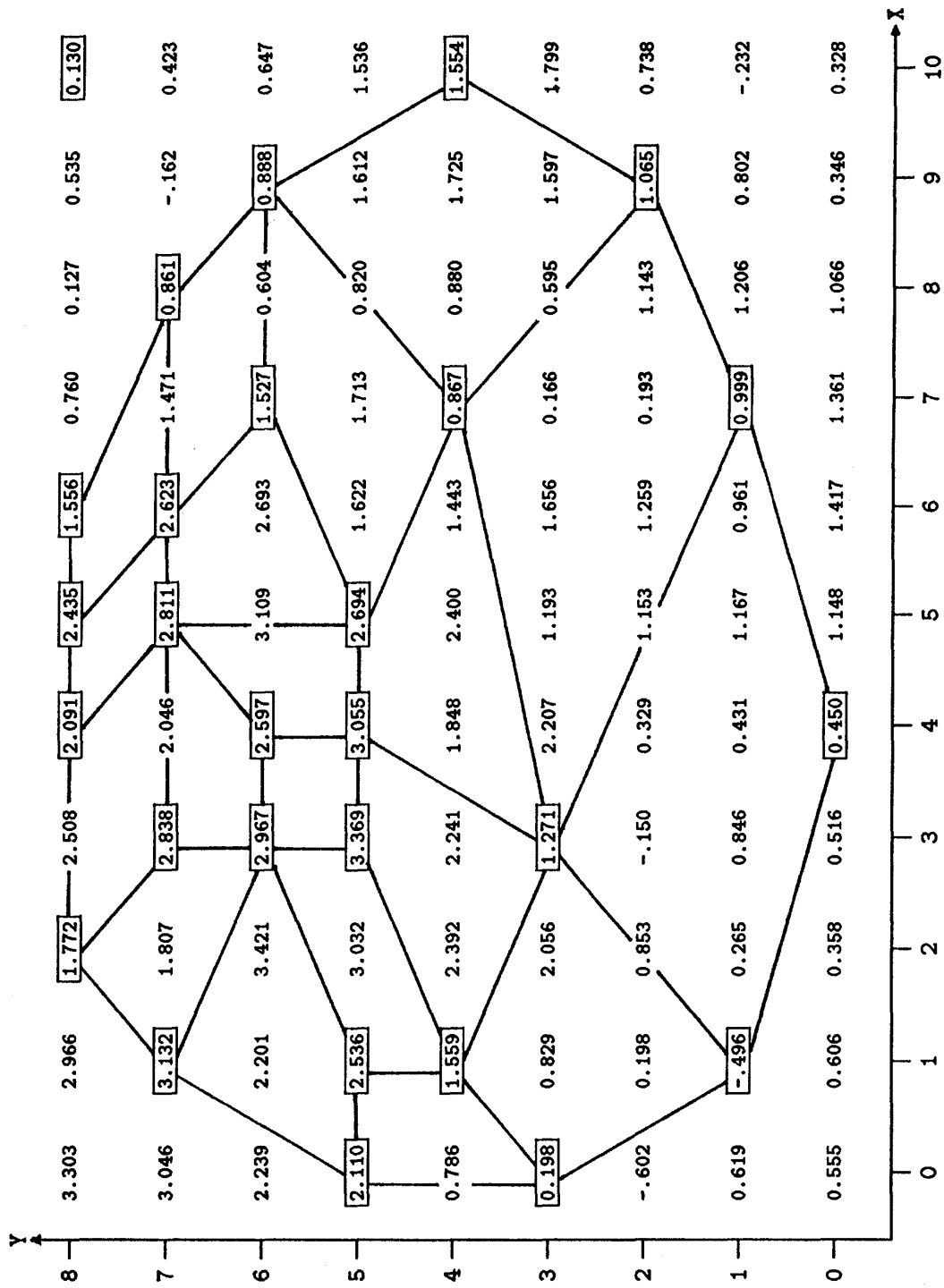


Figure 21. Finite element grid superimposed onto a point-centered finite difference grid.

### § 4.2.6 Generating Subregions at Higher Resolution

Sometimes it is desirable to have a portion of a previously generated random field represented at an increased level of detail. In many instances, this can be accomplished simply by specifying the coordinate positions of the extra points as in section 4.2.5. However, there are limits to the degree of increased resolution obtainable. If it is known in advance that areas of higher resolution will be needed, then the simulations can be designed to allow for that option later on.

To understand how to properly generate a higher resolution subregion, one should be thoroughly familiar with the details of the Turning Bands method, and in particular, the way it is implemented in this code. The line processes *must* be generated in *exactly* the same way as they were when the original field was created. Several different parameters control the generation of the line processes, the most obvious one being the number of Turning Band lines. When the default Turning Band parameters are used, this number is always set to 16. The other line process control parameters are calculated internally and must satisfy constraints arising from the geometry of the problem, the choice of covariance function, and other input specifications. The values of these internal parameters are listed at the end of the <NAME>.INP file which is created everytime TUBA is run. It is advisable to study subroutines DEFPAR and INTPAR to understand how these parameters are calculated. A constraint on the degree of resolution obtainable is imposed by the parameter UN which equals the discretization distance along the Turning Band lines. Points in the two-dimensional field must not be spaced closer than UN units apart, otherwise adjacent points in the two-dimensional field will receive the same projections from the lines and will therefore be indistinguishable.

#### Previous Examples

In section 4.2.5, we needed to insure that the internal parameter TBMX was the same in both of those runs in order to obtain identical values at the nodes of the finite element and finite difference grids. In section 4.2.4, our objective was to compare the areal average process on the 100 x 100 grid with the non-overlapping areal average process on the 20 x 20 grid. The FFT spectral method was used to generate the line processes and the number of harmonics (which equals the number of discrete points along each Turning Band line) required was 4096 for the 100 x 100 grid. If we would have accepted the default Turning Band parameters for the 20 x 20 grid case, the number of harmonics would only have been equal to 2048 (therefore the fields would not correspond). This is because of a constraint that prohibits<sup>†</sup> the line discretization distance,  $\Delta\zeta$ , (chosen initially to equal  $0.06\lambda$ ) from exceeding the grid spacing,  $\Delta x$ , which equals  $0.05\lambda$  in

---

<sup>†</sup> Only when the default Turning Band parameters are chosen.

the 100 x 100 case. When  $\Delta\zeta$  is decreased, the number of harmonics may have to be increased in order to satisfy a constraint on the maximum allowable frequency spacing. So, in order to *set* the number of harmonics at 4096 for the 20 x 20 case, we had to enter the Turning Band parameters manually. We also had to be careful to set the line process discretization distance,  $\Delta\zeta$ , equal to  $0.05\lambda$  in the 20 x 20 case. The proper Turning Band parameter values can be determined by examining the <NAME>.INP file for the 100 x 100 case. Note that the parameter values (e.g., number of Turning Band lines, number of harmonics, etc.) listed at the bottom of this file are the same for both the 100 x 100 and 20 x 20 cases.

### Subregion Generation Procedure

The easiest way to obtain the required Turning Band parameter values for subregion generation is to run TUBA as if the entire field were being generated on a high resolution grid; as soon as TUBA begins reporting its progress on calculating the projections, abort the program. Then print the <NAME>.INP file to get a listing of the parameter specifications that will be necessary for generating the high-resolution subregion. This is a handy ‘trick’ if it is known a priori that a high-resolution subregion will be needed. Even when this is not the case, it may still be possible to obtain the desired resolution; how much extra resolution is possible depends on how the original field was generated.

The Turning Band line discretization distance, UN, is one of the primary parameters controlling the amount of increased resolution one can obtain. When the default Turning Band parameters are chosen, TUBA calculates UN to be equal to  $\frac{1}{16}$ th the shortest correlation length, or, equal to the smallest spatial discretization,  $\Delta x$  or  $\Delta y$ . To determine the maximum amount of increased resolution possible for your case, divide the smallest spatial discretization distance listed at the bottom of the <NAME>.INP file by the Turning Band line discretization distance, also listed at the bottom of the <NAME>.INP file. The procedure is demonstrated in the following example.

Suppose a field with  $(x, y)$  dimensions equal to 128 by 64 length units was generated onto a 32 x 16 mesh. The <NAME>.INP listing file for such a field is shown below. Note that the Turning Band line discretization distance is  $\frac{1}{16}$ th the correlation length and that the maximum resolution possible equals  $\Delta X/UN = \frac{4}{0.625} = 6.4$ . We choose to generate a subregion of this field at six times the resolution. The subregion geometry specifications are shown in the GSUB.INP listing file; note that a point-centered grid is specified for the subregion in order that grid points of the coarse mesh coincide with those of the fine mesh(see Figure 22). The Turning Band parameters are entered manually for the subregion data, the values taken from the GBAC.INP listing file. The resulting fields are plotted in Figure 23.

Input Data for Coarse Grid Data [GBAC.INP]

```

      2          1=(x,y) Locations, 2=Gridded output
      2          1=Point Centered, 2=Block Centered
128.0000  64.0000 Maximum X and Y Field Dimensions
      32          16 Number of Nodes-X and Nodes-Y
NONE      Mask Filename
      1          1=Normal, 2=exp(X), 3=10**(X)
      2          0=User,1=Exp,2=Gauss,3=Bes1,4=Telis,5=GC
      .0000      1.0000 Desired Mean and Variance
10.0000  10.0000 X and Y Direction Correlation Lengths
      1          1=Default TBM Parameters, 2=Enter Manually
GBAC.DAT  Output Data Filename
      1          1=Unformatted, 2=Formatted Output
      2          1=Single Write Statement, 2=Line at a Time
      2          1=Marsaglia URNG, 2=Machine Indep URNG
73333330  Seed for Random Number Generator
      1          Number of Realizations to be Simulated

! Field origin relative to TBM origin =      .0000      .0000
! Number of Turning Band Lines Equals =      16
! The Maximum Turning Band Line Length =      143.1084
! Turning Band Line Discretization Lgth =      .6250      ← note value
! Maximum Frequency for the Spectrum =      100.5310
! Number of Harmonics for the Spectrum =      1024
! Frequency Spacing in Spectral Domain =      .0982
! Spatial Discretizations, DELX & DELY =      4.0000      4.0000 ← note value
! No Pnts/correlation Length in X,Y Dir =      2.5      2.5
! Approx Number of Independent Samples =      20.5
! The Sample Mean =      -.2643
! Sample Variance =      .9897

```

Input Data for Fine Grid Data [GSUB.INP]

```

      2          1=(x,y) Locations, 2=Gridded output
      1          1=Point Centered, 2=Block Centered      ← point centered
88.0000  40.0000 Maximum X and Y Field Dimensions
      133         61 Number of Nodes-X and Nodes-Y
NONE      Mask Filename
      1          1=Normal, 2=exp(X), 3=10**(X)
      2          0=User,1=Exp,2=Gauss,3=Bes1,4=Telis,5=GC
      .0000      1.0000 Desired Mean and Variance
10.0000  10.0000 X and Y Direction Correlation Lengths
      2          1=Default TBM Parameters, 2=Enter Manually ← manual entry
      16         Number of Turning Band Lines
      .6250      TBM Line Discretization Distance
      1024       Nbr of Harmonics for Discretizing Spectrum
20.0000  12.0000 Field Origin Relative to TBM Origin
143.1084 Maximum Turning Band Line Length
GSUB.DAT  Output Data Filename
      1          1=Unformatted, 2=Formatted Output
      2          1=Single Write Statement, 2=Line at a Time
      2          1=Marsaglia URNG, 2=Machine Indep URNG
73333330  Seed for Random Number Generator
      1          Number of Realizations to be Simulated

! Field origin relative to TBM origin =      20.0000      12.0000
! Number of Turning Band Lines Equals =      16
! The Maximum Turning Band Line Length =      143.1084
! Turning Band Line Discretization Lgth =      .6250
! Maximum Frequency for the Spectrum =      100.5310
! Number of Harmonics for the Spectrum =      1024
! Frequency Spacing in Spectral Domain =      .0982
! Spatial Discretizations, DELX & DELY =      .6667      .6667
! No Pnts/correlation Length in X,Y Dir =      15.0      15.0
! Approx Number of Independent Samples =      8.8
! The Sample Mean =      .0166
! Sample Variance =      1.1305

```



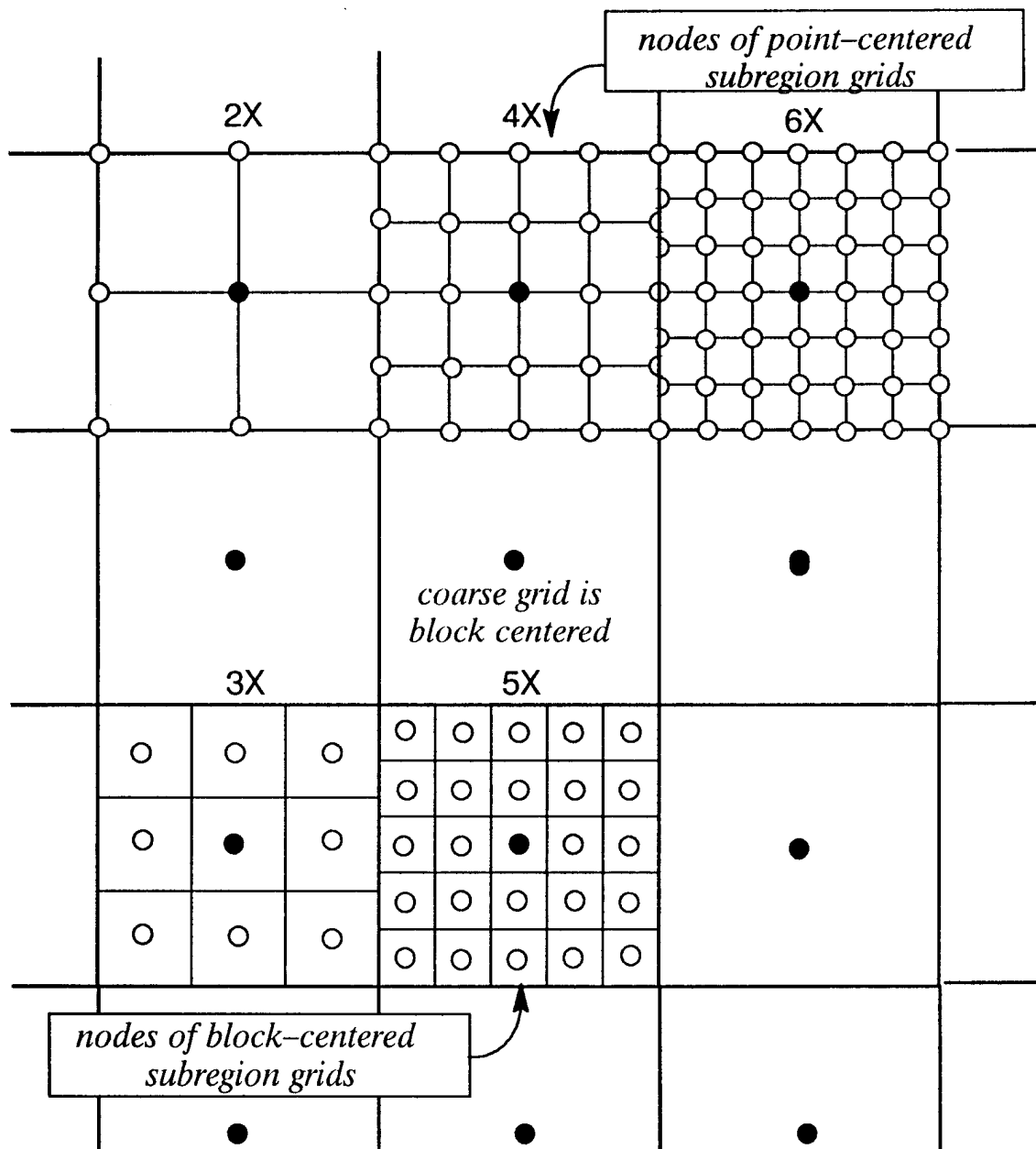


Figure 22. Schematic showing that, to achieve alignment between two different meshes where the coarse mesh is block centered, a point-centered grid is needed for even increases in resolution while a block-centered grid is needed for odd increases in resolution. The origin of both grids is assumed to coincide.

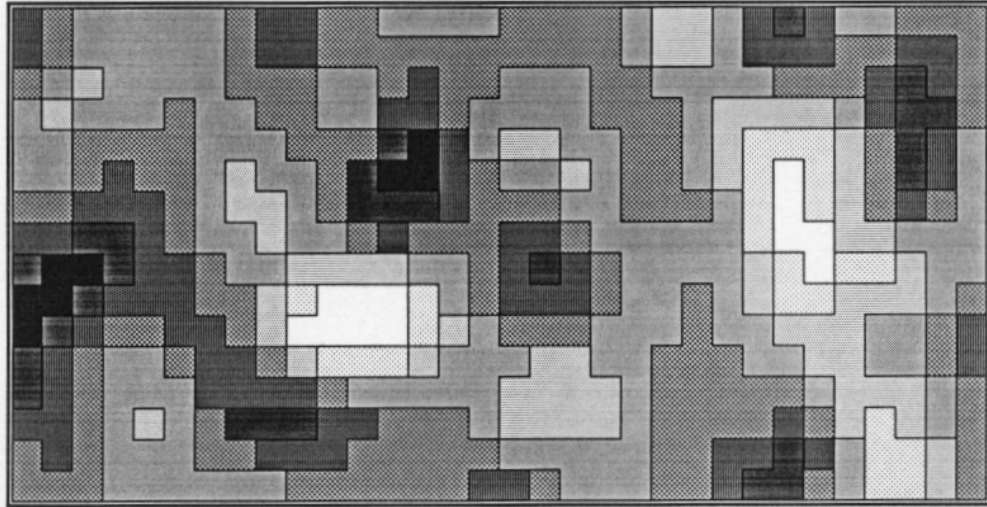


Figure 23a. A Gaussian field generated onto a  $32 \times 16$  mesh.

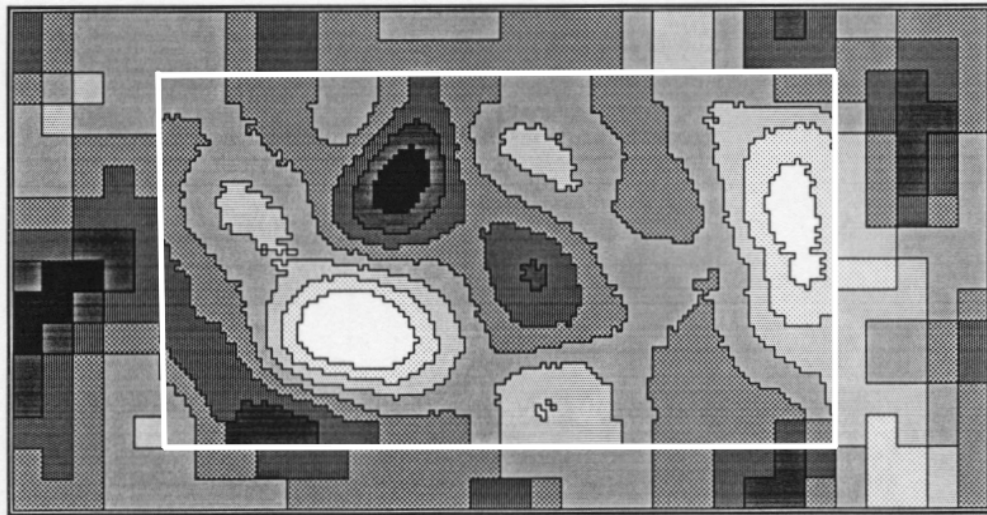


Figure 23b. The same Gaussian field shown in Figure 23a with a subregion (white inset) generated at six times the original resolution. The coarse  $32 \times 16$  field was expanded six fold and the subregion data was merged into it to illustrate the result more clearly.

## § 4.3 Some Practical Aspects Illustrated

In this section we present some examples that illustrate some of the ideas discussed in sections 2.1, 2.3 and 3.2, and provide examples that demonstrate the versatility of the code. The manner in which one chooses to display the generated fields is discussed and alternatives are presented. Finally, some important remarks concerning the generation of very large fields is noted.

### § 4.3.1 Sample Statistics Versus Target Statistics

Note that the target mean and variance (zero and one respectively) of the fields generated in section 4.2.1 is not attained with great accuracy in any one of these examples. This is because these synthetic fields represent only a small portion of the “true” underlying random field; for example, the exponential and Gaussian fields span only 5 correlation lengths (field width=100,  $\lambda = 20$ ). True independence between field values occurs only at separation distances of say, 2 or 3 correlation lengths, therefore, there were only a small number of independent samples ( $\approx 5$  or 6) upon which to calculate the mean and variance statistics. Averaging these statistics over an ensemble of simulations would show convergence toward the theoretical statistics (*Mantoglou and Wilson [1982]*). On the other hand, since the generation method is ergodic, statistics based on a single large realization should show improved accuracy. To verify this, we regenerated the Telis field shown in Figure 11, but specified the correlation lengths equal to 2.0 instead of 27.0. This has the effect of increasing the size of the realization so that now it contains more than 100 times as many independent samples as the original Telis field. The sample statistics listed at the bottom of the TELIS-BIG.INP file shown below, now correspond very well with the target or theoretical statistics.

Input file for “large” Telis field [ TELIS-BIG.INP ]

```

          2          1=(x,y) Locations, 2=Gridded output
          2          1=Point Centered, 2=Block Centered
    100.0000 100.0000 Maximum X and Y Field Dimensions
          100          100 Number of Nodes-X and Nodes-Y
NONE                                     Mask Filename
          1          1=Normal, 2=exp(X), 3=10**(X)
          4          0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC
          .0000    1.0000 Desired Mean and Variance
          2.0000    2.0000 X and Y Direction Correlation Lengths
          1          1=Default TBM Parameters, 2=Enter Manually
TELIS-BIG.DAT Output Data Filename
          1          1=Unformatted, 2=Formatted Output
          2          1=Single Write Statement, 2=Line at a Time
          2          1=Marsaglia URNG, 2=Machine Indep URNG
          76548921 Seed for Random Number Generator
          1          Number of Realizations to be Simulated
```

```

! Field origin relative to TBM origin =      .0000      .0000
! Number of Turning Band Lines Equals =      16
! The Maximum Turning Band Line Length =    141.4214
! Turning Band Line Discretization Lgth =    .1250
! Discretization Dstnce for MA Process =     .1000
! Number of Output Pnts Along the Line =    1132
! Spatial Discretizations, DELX & DELY =     1.0000      1.0000
! No Pnts/correlation Length in X,Y Dir =     2.0        2.0
! Approx Number of Independent Samples =     625.0
!
!                               The Sample Mean =      .05722
!                               Sample Variance =      1.01328 ← Note improved
                                                                    sample statistics

```

### § 4.3.2 Representing Spatial Variability Patterns

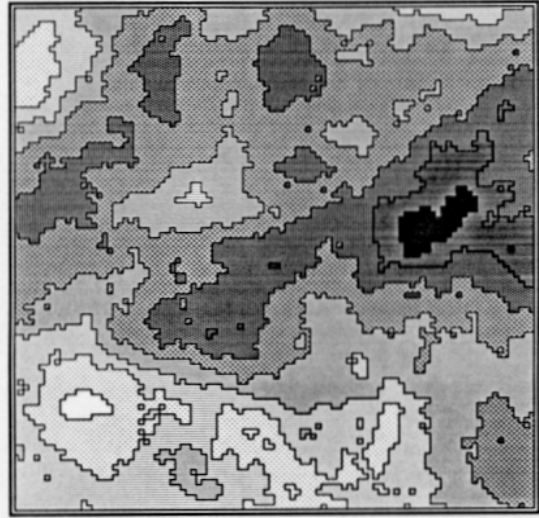
The manner in which data are represented plays an important role in determining how the data will be interpreted. In this section, we present alternate ways of displaying the data to point out some subtle but important aspects of data representation and interpretation. We also provide an example that demonstrates the utility of the “mask file” option.

#### Grid Resolution

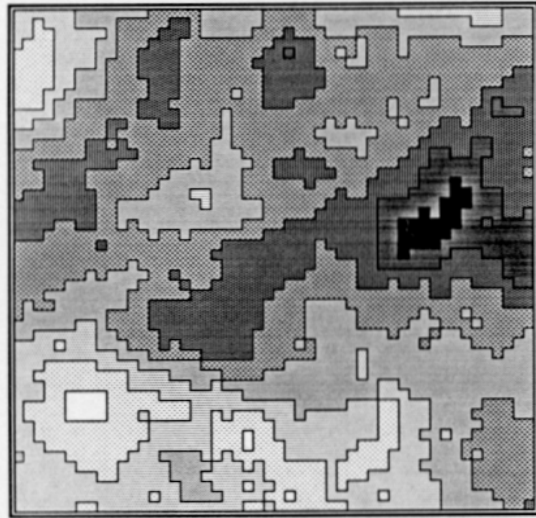
In the first example, a random field is generated at three different levels of resolution: five, ten, and 20 points per correlation length. The fields were generated using the Bessel covariance function and are plotted in Figure 24. The high resolution field was generated onto a  $100 \times 100$  mesh while the low and medium resolution fields were generated onto grids of size  $25 \times 25$  and  $50 \times 50$  respectively. The grids for these latter two fields were expanded to  $100 \times 100$  (for plotting purposes) by duplicating mesh points. All three fields represent the same physical dimensions, but there is a marked difference in the clarity with which they reflect the behavior of the “true” underlying random field. A numerical modeler may tend to view the model input data as being representative of the actual field, and thus conceptualize the variability in the data as that shown in the high-resolution field on the right in Figure 24. What the computer model typically “actually sees,” however, is data that is more accurately represented by the plot on the left in Figure 24.

#### Alternate Displays

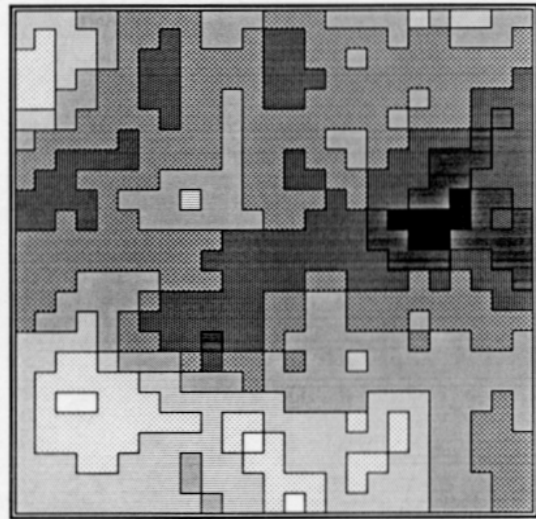
The second example illustrates some alternative ways of plotting the synthetic fields. Figure 25 shows a Gaussian field plotted using regular contour lines (top) and shaded “terraces” (bottom). The spatial variability patterns of this smooth Gaussian field exhibit similar behavior regardless of the algorithm used to plot the data. A “noiser” exponential field is shown in Figure 26. Here, the contour line approach presents little



20 points/correlation length



10 points/correlation length



5 points/correlation length

Figure 24. A random field generated at three different levels of resolution.

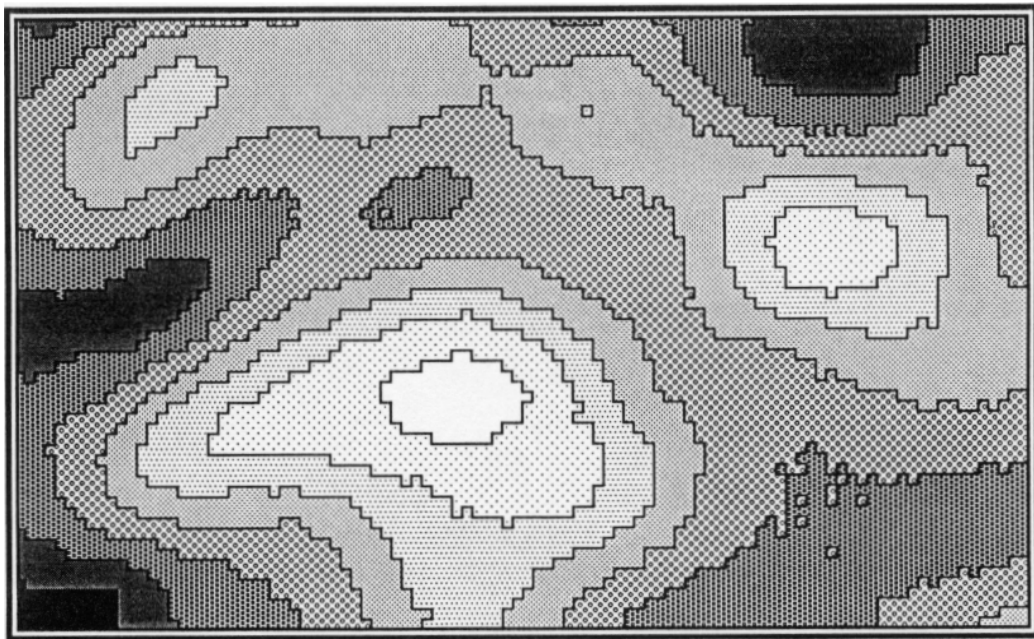
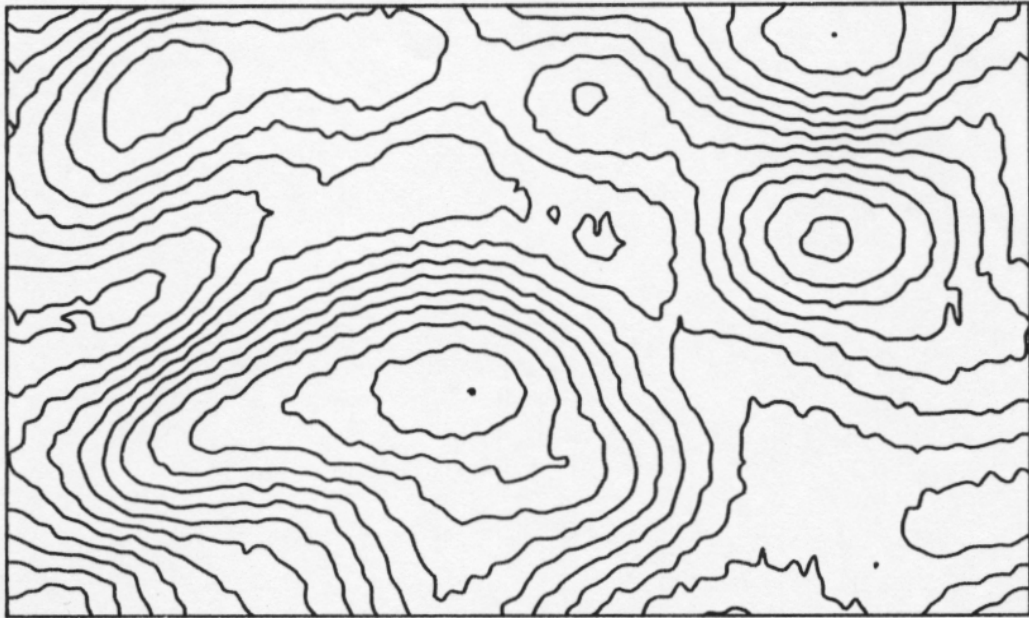


Figure 25. A Gaussian field plotted using contours and shaded terraces.

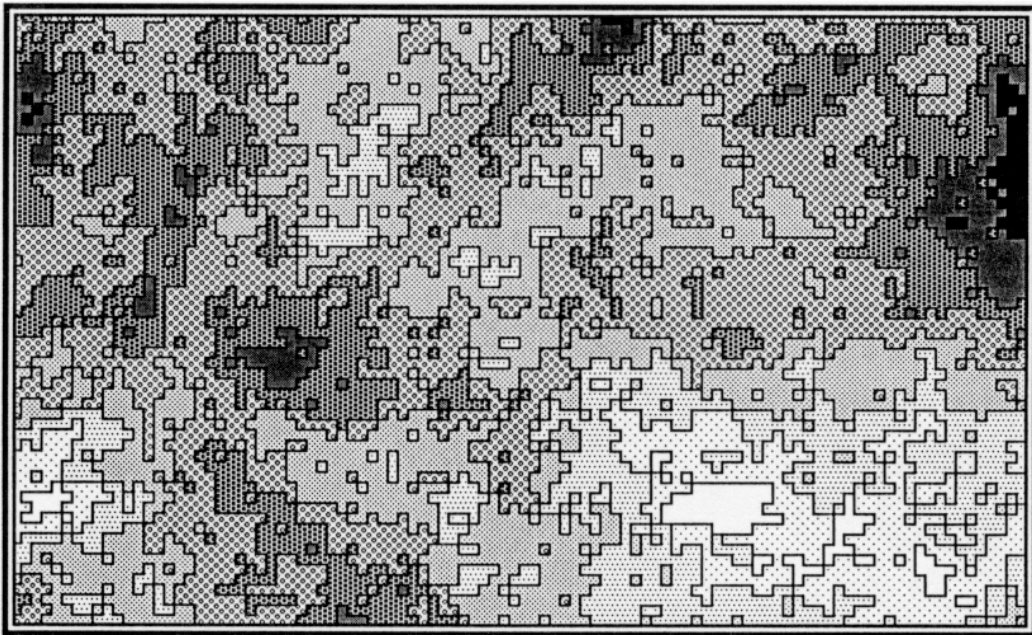
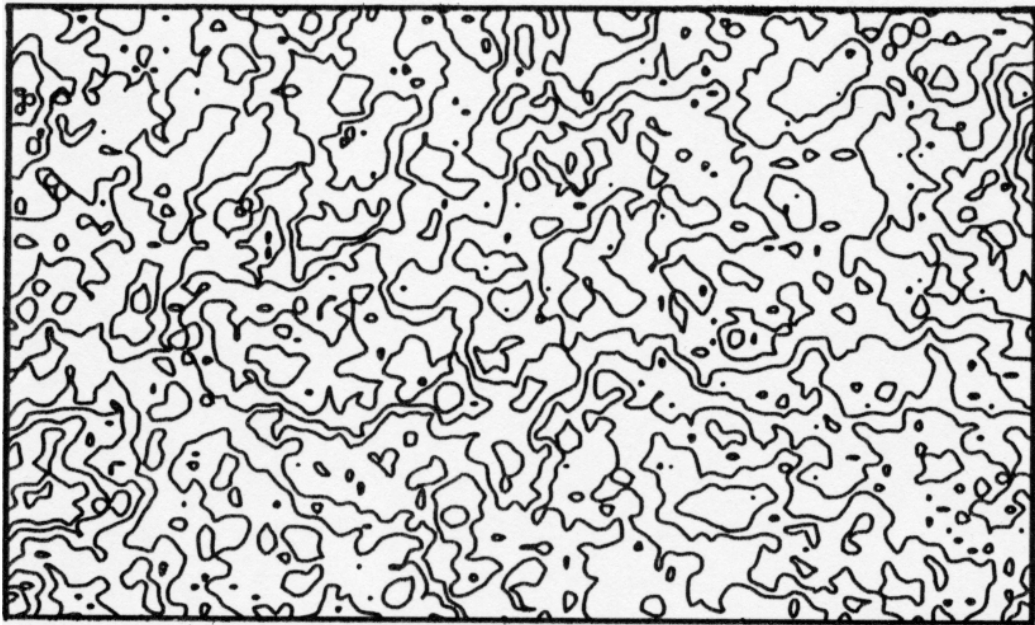


Figure 26. An exponential field plotted using contours and shaded terraces; the topography is easier to interpret from the shaded representation.



information, while the shaded plot displays information concerning the highs and lows in a manner that is readily comprehensible. Both the smooth Gaussian field and the noisy exponential field are displayed as surfaces in three-dimensions in Figure 27. Both are scaled identically. The exponential field is plotted using profiles while the Gaussian field is plotted using terraced contours. The exponential field exhibits short-range fluctuations relative to the Gaussian field as one would expect from the spectral distribution function curves (Figure 4).

### Log-normal Fields

In all of the examples discussed so far, we have examined the spatial variability patterns of normally distributed processes. In many applications we are interested in a process which is log-normally distributed; for example, in hydrology an aquifer's hydraulic conductivity field is assumed to be a log-normally distributed process. In that case, we generate the logarithm of hydraulic conductivity and then exponentiate the field to obtain the actual hydraulic conductivity values. The shaded plots shown in this manual were generated by dividing the range of (normally distributed) values into 8 equally-spaced intervals and assigning a shade pattern to each interval. When the field is exponentiated, the distribution becomes skewed rather than normal, with the predominance of values at the low end of the range. When this log-normal field is plotted in the same manner, i.e., by dividing the skewed distribution into an histogram of eight equal increments and assigning a shade pattern to each, the predominance of low values becomes readily evident. This is illustrated in Figure 28 where the same Gaussian field is plotted, except that the data were exponentiated in the final step of field generation. The lower plot shows profiles of the lognormal field in three-dimensions; the vertical scale on the left of that plot shows the eight data intervals that were used to create the plot shown at the top of Figure 28. The shaded plot might lead one to erroneously conclude that the data are more homogeneous than it actually is. The figure illustrates the importance of understanding what the data represent (e.g., normal/lognormal) and what the plotting algorithm is doing.

### The Mask File Option

The last example is included to illustrate the use of the mask file option. This option allows the user to input a matrix that controls which grid points the field values will be generated on; the mask file must have the same grid dimensions as the output field. Random field calculations will not be performed for any grid point at which the mask file contains an exact zero, instead, the output value will be an exact zero. A mask file might typically contain only 1's and 0's (1=yes, do generate a value for the random field



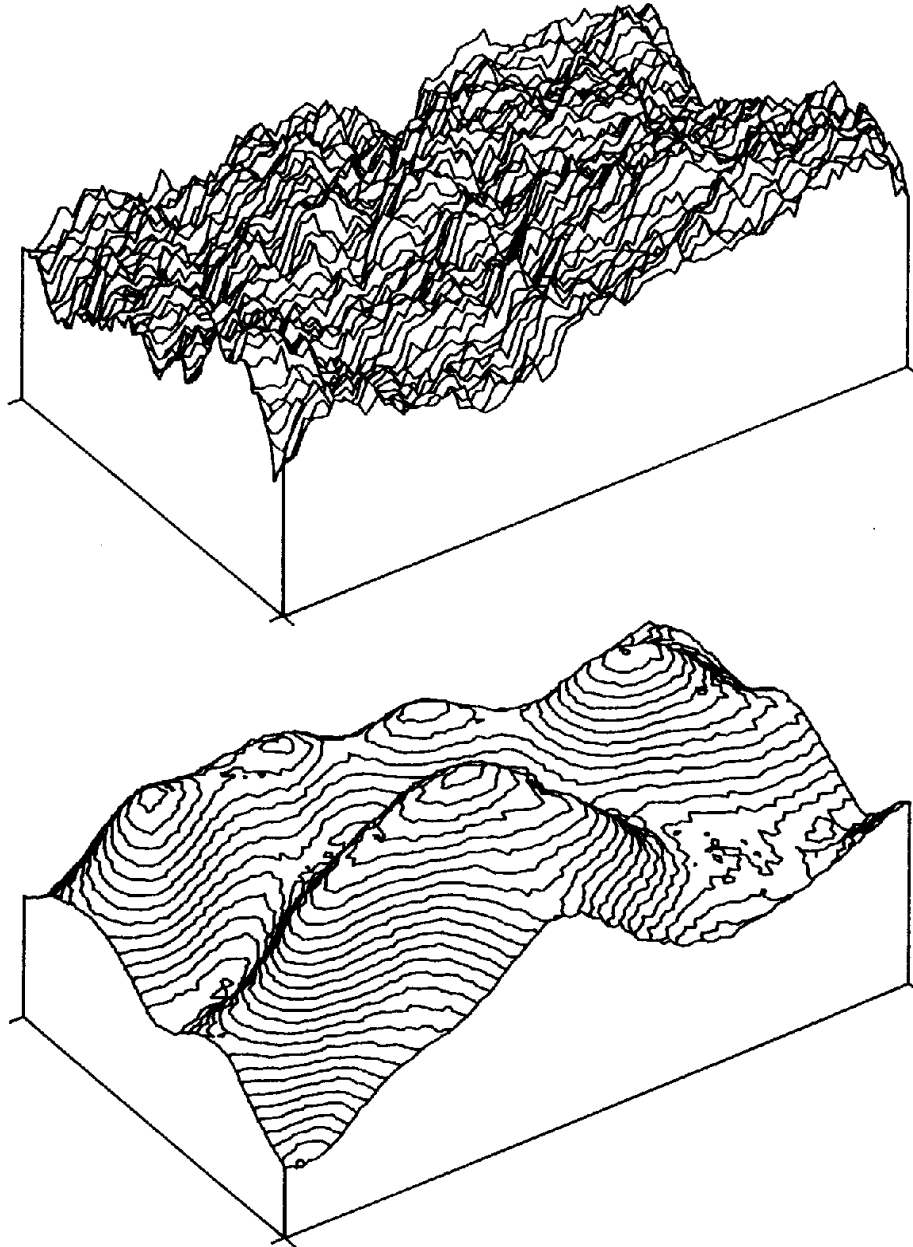


Figure 27. An exponential field (top) plotted using profiles and a Gaussian field (bottom) plotted using terraced contours.

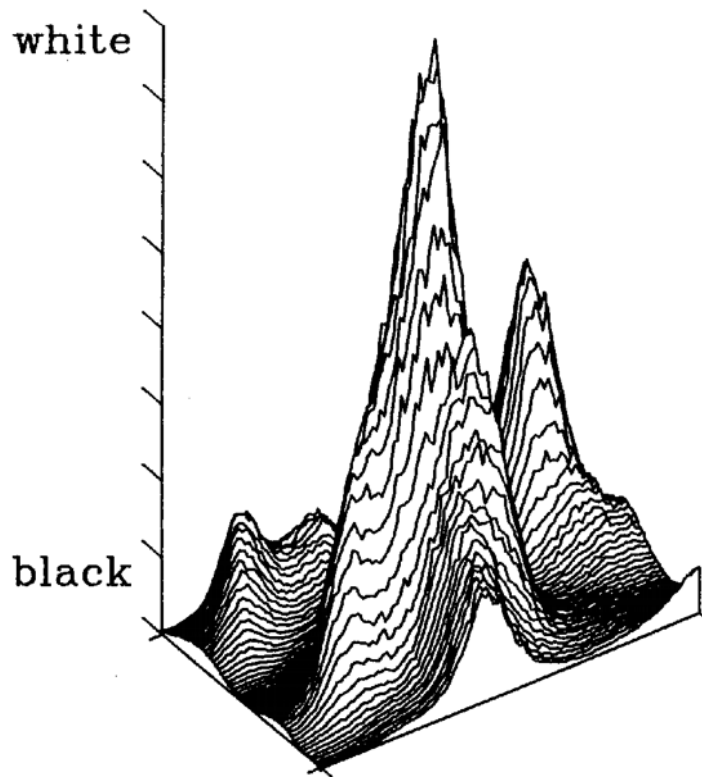
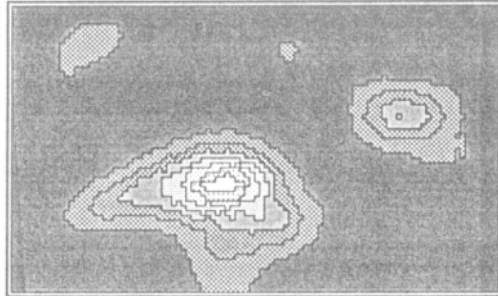


Figure 28. The exponentiated version of the Gaussian field shown in Figures 25 and 27 plotted in two different ways.

at this location, 0=no, do not generate a value here), although a matrix that contains exact zeros and any non-zero values will work. *The mask file must be a formatted ASCII file*, for example, a file containing 1’s and 0’s will work fine; the mask file data are read using a free-format read statement (`READ(LU,*)`).

The mask file option may be useful for modelers who are simulating some process in a domain that is bounded by an irregular geometry. In this example, a Gaussian field was generated onto a rectangular mesh using a mask file to control the locations at which the field values were output. The mask file defines the shape of the areal extent of an aquifer in the Avra Valley, Arizona. The resulting field is plotted in Figure 29; the gray shaded area, between the plot border and the interior heterogeneous region, represents the areas where exact zeros were output.

### § 4.3.3 Using the Spectral Method of Shinozuka and Jan

The “smoothness” of the Gaussian field compared to the exponential field (see Figures 8 and 10) was explained in section 4.2.1 in terms of the spectra of these covariance models, i.e., the Gaussian spectrum lacks high frequency content. In this example, we generate an exponential field but “chop off” the high frequencies to illustrate the role those frequencies play in affecting the spatial character of the field, and to provide an example where the line processes are generated via the method of *Shinozuka and Jan* [1972].

We wish to compare this result with the exponential field generated in section 4.2.1 (Figure 10), therefore the field dimensions, the number of nodes, and the correlation length parameters are all set the same as in that example. However, instead of generating the one-dimensional line processes using frequencies as high as 100 cycles/length (cpl), as is recommended in Table 3, we truncate the spectrum at a maximum frequency of  $F_{MAX} = 10$  cpl. Figure 4 shows that 90% of the variance is contained at frequencies less than or equal to 10 cpl for the exponential model; the remaining 10% corresponds to the high frequency content which produces the “business” in the field shown in Figure 10. Table 3 shows that a frequency spacing of  $\Delta k \leq 0.05$  is recommended for accurately representing the spectrum of the exponential model. Thus, choosing the number of harmonics,  $M = 200$ , would be recommended for this case in order that  $\Delta k = \frac{F_{MAX}}{M} = 0.05$ . However, the exponential field shown in Figure 10 was generated with  $\Delta k = 0.03068$  (see the `EXP01.INP` file, section 4.2.1), therefore, we chose  $M = 326$  such that  $\Delta k = \frac{F_{MAX}}{M} = 0.03067$  in order to match the frequency spacing in that example. The input file is shown above the plotted result (Figure 30); note that without the high frequencies represented, the spectral character of the generated field more closely resembles the Bessel field shown in Figure 9 which has 99% of its spectrum contained at frequencies less than 10 cpl.

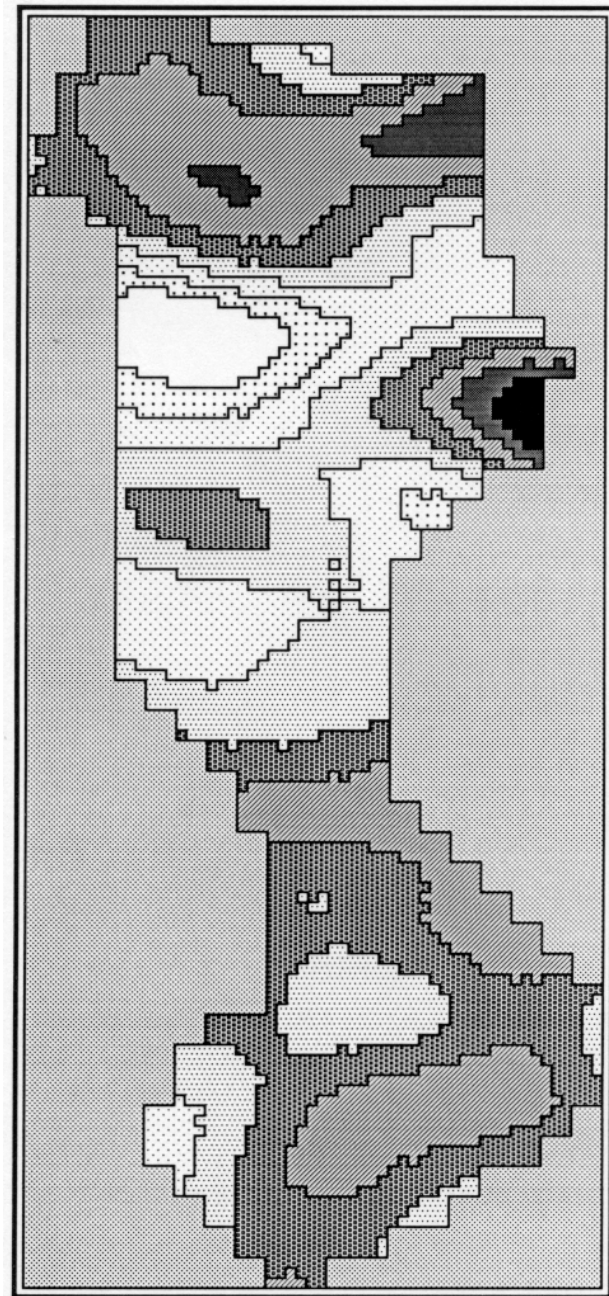


Figure 29. A Gaussian field generated onto a domain bounded by an irregular geometry. TUBA's "mask file" option was used to inhibit field generation outside the heterogeneous region.

Input file for Shinozuka and Jan exponential field [EXPO-SAJ.INP]

```

      2          1=(x,y) Locations, 2=Gridded output
      2          1=Point Centered, 2=Block Centered
100.0000 100.0000 Maximum X and Y Field Dimensions
      100          100 Number of Nodes-X and Nodes-Y
NONE          Mask Filename
      1          1=Normal, 2=exp(X), 3=10**(X)
      1          0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC
      1          1=Point Process, 2=Areal Average Process
.0000      1.0000 Desired Mean and Variance
20.0000 20.0000 X and Y Direction Correlation Lengths
      2          1=Default TBM Parameters, 2=Enter Manually ← manual entry
      16          Number of Turning Band Lines
      1.0000      TBM Line Discretization Distance
      326          Nbr of Harmonics for Discretizing Spectrum ← ≠ 2n, therefore, method of
10.0000      Max Frequency for Truncation of Spectrum          Shinozuka and Jan is used
.0000      .0000 Field Origin Relative to TBM Origin
141.4214      Maximum Turning Band Line Length
EXPO-SAJ.DAT Output Data Filename
      1          1=Unformatted, 2=Formatted Output
      2          1=Single Write Statement, 2=Line at a Time
      2          1=Marsaglia URNG, 2=Machine Indep URNG
57756341      Seed for Random Number Generator
      1          Number of Realizations to be Simulated

! Field origin relative to TBM origin =      .0000      .0000
! Number of Turning Band Lines Equals =      16
! The Maximum Turning Band Line Length =      141.4214
! Turning Band Line Discretization Lgth =      1.0000
! Maximum Frequency for the Spectrum =      10.0000
! Number of Harmonics for the Spectrum =      326
! Frequency Spacing in Spectral Domain =      .0307

```

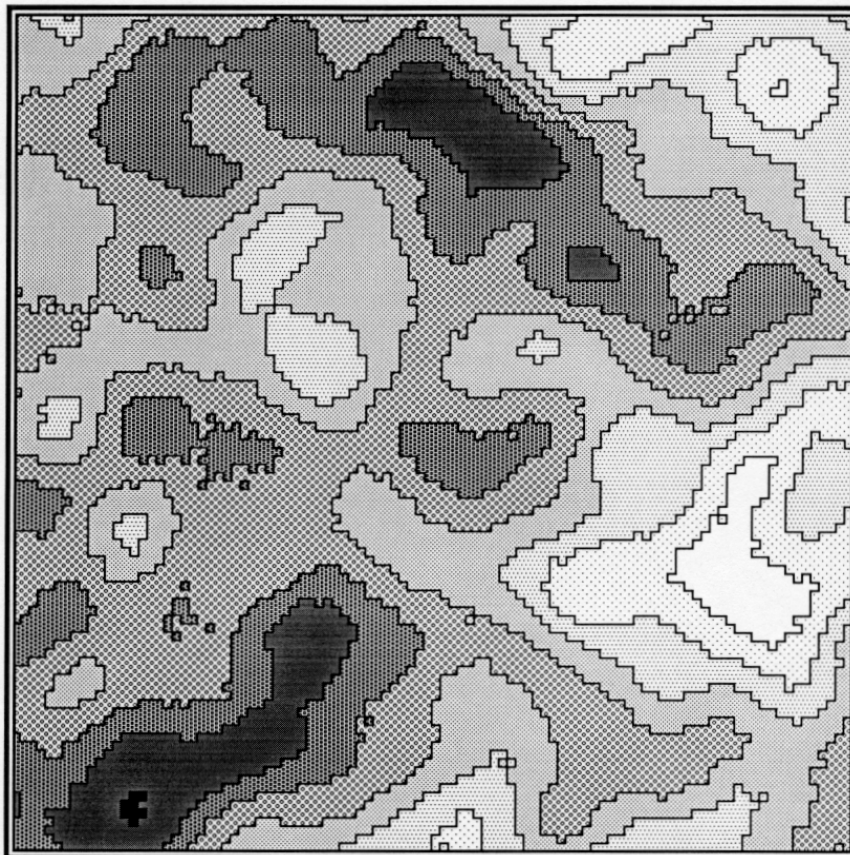


Figure 30. An exponential random field generated using the method of Shinozuka and Jan.

#### § 4.3.4 Hydrology Application – A Layered System

Most of the random fields presented thus far might be used to represent the spatial distribution of certain properties of earth materials such as ore grade or permeability. In hydrology, the properties of interest may take on a distinctly different character in the vertical direction compared to the horizontal plane due to the manner in which the sediments were deposited. For example, *Peterson and Wilson* [1988] discusses the vertical variation in the hydraulic properties of alluvial aquifers associated with underfit streams (streams that appear too small for the valley in which they flow). Such streams arise from changes in climatic conditions over geologic time where past discharges were much higher than is observed today.

During the earlier wet periods (e.g., the last glaciation), the high flows in streams and rivers were capable of transporting and depositing large coarse-grained materials; in recent times with milder climatic conditions, stream flows have decreased and the suspended and bed loads have become more fine grained. As a result of this depositional history, the shallow sediments tend to be highly stratified materials composed of fine-grained sands interbedded with silty and clayey flood plain deposits, while the deeper sediments are made up of coarse well-sorted materials. The aquifer's hydraulic properties are therefore likely to be more homogeneous, isotropic and transmissive at depth than near the surface where the finer grained materials are less permeable, the stratification induces anisotropy, and the system is in general, more heterogeneous.

In this example, we attempt to simulate the material property variations of such a system by generating a synthetic two-dimensional vertical cross-section random field possessing the hydraulic characteristics described above. The vertical profile consists of three layers, hence three different random fields were generated and “stacked” on top of each other.

The increase in homogeneity with depth was represented by generating the fields with different covariance functions – the Telis, the Bessel, and the Gaussian covariance models respectively, from top to bottom. Very large fields were generated for each of these covariance models and subregions were extracted from each such that the spatial variability patterns matched fairly well along the interface between layers. A moving-average box filter was passed along the interface to provide a smooth transition of the data values from one layer to the next. The Telis field was generated with an  $x:y = 10:1$  anisotropic correlation structure while the Bessel and Gaussian fields were generated using ratios of 4:1 and 1:1 respectively.

The increase in transmissivity with depth was accomplished by generating the fields with a linear trend in their means; the dark shades near the top represent low transmissivity values while the lighter shades near the bottom indicate higher transmissivity values. The resulting field, illustrated in Figure 31 below and on the cover of this document, appears to exhibit the types of features we were trying to represent.

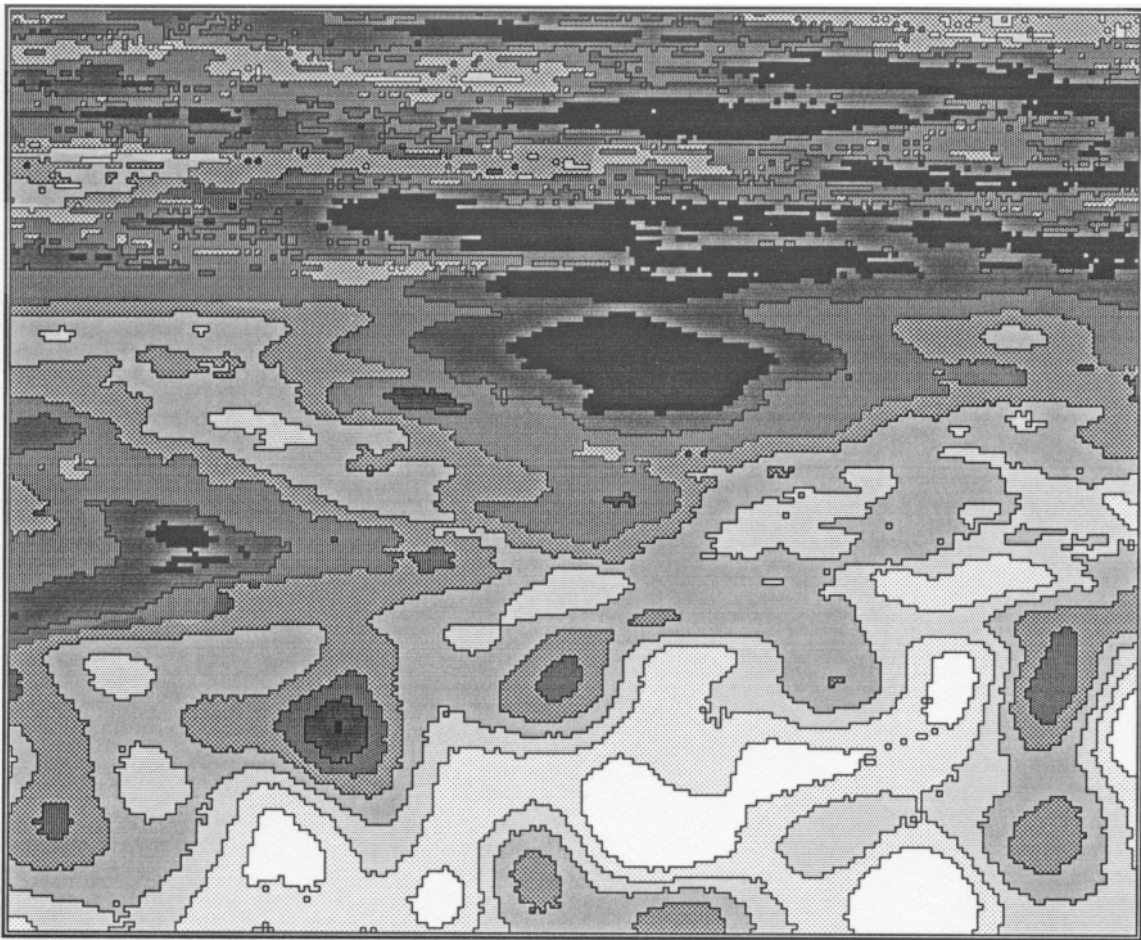


Figure 31. A random field representing variations in the hydrologic properties of a layered subsurface environment.

### § 4.3.5 Designing Christmas Cards with TUBA

A common and legitimate concern of persons reviewing the results of a computer modeling study is whether the computer code was properly applied. Misapplication of the code produces inaccurate or incorrect results which leads to misinterpretations and erroneous conclusions. This is especially unfortunate when the data or procedural errors are subtle enough to go unnoticed and the ensuing (defective) analysis is used for decision making. In this section, we demonstrate, in a not so subtle way, what can happen when TUBA is not applied properly; this is an excellent example of the garbage in  $\rightarrow$  garbage out principle. The example also serves to illustrate which areas of the two-dimensional plane not to position the Turning Bands origin.

Suppose, for the sake of a story, a Telis field had been generated onto a point-centered grid, 100 cells on a side; the spatial variability patterns of such a field is shown in Figure 11. Don Dweeb, the computer programmer, was instructed to generate the field values in the center of the grid cells (i.e., a block-centered grid) and to use 18 Turning Band lines instead of 16. The Turning Band parameters must be entered manually in order to use anything other than 16 lines. Don had a copy of the <NAME>.INP file that was used to generate the original field in front of him so he could see what other Turning Band parameter values to enter; that's when his troubles began ...

Don Dweeb was a bit of a slob; his shirt tail would always be hanging out, he was unkempt and had fast-food packaging strewn all about his desk. Because Don was mostly nocturnal, he had trouble reading the <NAME>.INP file (mostly because he was too lazy to turn on all the lights). Furthermore, he had just finished eating a candy bar, and his blood sugar was "up and running," causing him to be somewhat spastic, whereupon he accidentally entered 8 instead of 18 for the number of Turning Band lines. Don was confused about the difference between point and block-centered grids and must have been thinking about "block-centered grids" with the "node in the center" when he specified the Turning Bands origin to be at the center of the grid (output field). And probably as a result of his fingers and the keyboard being sticky from melted chocolate, he accidentally hit the wrong keys and entered 18 for the moving-average process line discretization length; a correct value for this parameter would have been 1.0. The result of Don's madness is illustrated in Figure 33; it demonstrates clearly the importance of understanding how to properly apply the code. As it turned out, all was not lost, for when Don's mentor returned and saw the result, she rejoiced at Don's "genius" for giving her the idea to use fractal analysis in her work.

The symmetry in the patterns in this figure arises from the fact that the line processes are generated in one direction only, that is, outward from the origin in, say, the  $\phi$



direction. The exact same line process will be found in the  $\phi + 180$  degrees direction. When entering the Turning Band parameters manually, TUBA asks for the location of the output field origin relative to the Turning Bands origin; be careful not to position the field such that the Turning Bands origin is anywhere inside the output field domain.

Figure 32 shows those areas in the two-dimensional plane where it is okay to position the Turning Bands origin. Note that the Turning Bands origin may be placed on the boundary of a grid at only four points (the corners of the grid).

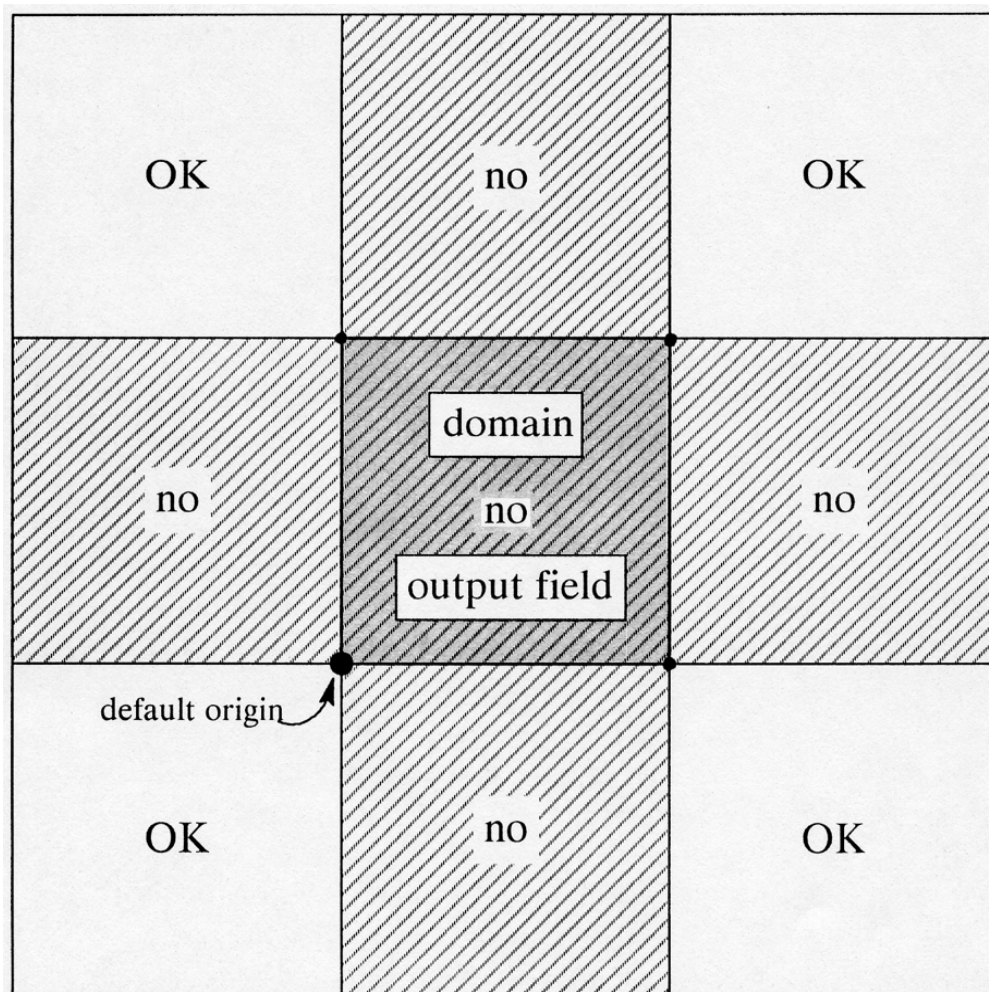


Figure 32. Cross-hatched areas are zones in the two-dimensional plane where the Turning Bands origin should not be placed.

Input file for Don Dweeb's Experiment [DON.INP]

```

      2          1=(x,y) Locations, 2=Gridded output
      2          1=Point Centered, 2=Block Centered
100.0000 100.0000 Maximum X and Y Field Dimensions
      100      100 Number of Nodes-X and Nodes-Y
NONE      Mask Filename
      1          1=Normal, 2=exp(X), 3=10**(X)
      4          0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC
      .0000    1.0000 Desired Mean and Variance
      50.0000 50.0000 X and Y Direction Correlation Lengths
      2          1=Default TBM Parameters, 2=Enter Manually
      8          Number of Turning Band Lines
      1.0000    TBM Line Discretization Distance
      18.0000    Discretization Distance for MA Process ← blunder
     -50.0000 -50.0000 Field Origin Relative to TBM Origin ← blunder
     142.0000    Maximum Turning Band Line Length
DON.DAT    Output Data Filename
      1          1=Unformatted, 2=Formatted Output
      2          1=Single Write Statement, 2=Line at a Time
      2          1=Marsaglia URNG, 2=Machine Indep URNG
     68416790    Seed for Random Number Generator
      1          Number of Realizations to be Simulated

! Field origin relative to TBM origin =   -50.0000   -50.0000
! Number of Turning Band Lines Equals =           8
! The Maximum Turning Band Line Length =    142.0000
! Turning Band Line Discretization Lgth =       1.0000
! Discretization Dstnce for MA Process =       18.0000
! Number of Output Pnts Along the Line =       142
! Spatial Discretizations, DELX & DELY =        1.0000    1.0000

```

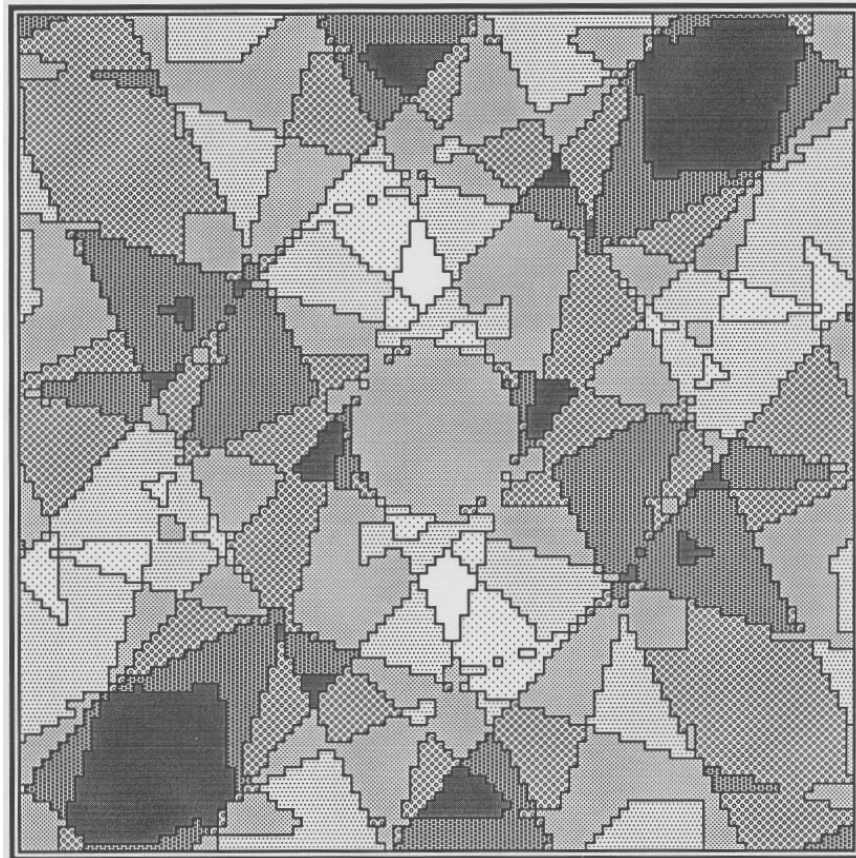


Figure 33. A random field generated by a mad scientist.

### § 4.3.6 An Important Note About Generating Large Fields

During the period when this documentation was being compiled and modifications to the code were being made, we conducted numerous experiments to test the new version of the code. Most of the synthetic fields we generated were small, spanning only 5 or 10 correlation lengths. However, we also generated fields spanning many correlation lengths and found some peculiar results. These large fields exhibited linear features oriented at different angles, crossing over each other throughout the field. An example is shown in Figure 34 in which an exponential field, 200 correlation lengths on a side, is plotted. The “lineaments” can be observed best by examining the plot from a view angle approximately 20 degrees above the plane of the plot and rotating it around. Initially, we thought there was an error in the coding, or a problem with the input specifications given when running the code. However, we could not find any coding errors and the input specifications, although not unique, appeared to be correct. Through experimentation we noticed that the “lineament effect” was attenuated as the number of lines was increased. Conversely, this effect is accentuated when the number of lines is reduced. Figure 35 shows an exponential field of the same size, generated using only 4 Turning Band lines; the lineaments in this figure are much more noticeable.

These lineaments are caused by a truncation error of the Turning Bands method due to an insufficient number of lines. *Mantoglou and Wilson* [1982] investigated the errors and found that, for the methods used in TUBA (evenly spaced lines on the unit circle), the covariance error,  $E_c$ , is given by

$$E_c = \beta \frac{\pi \sigma^2 K b r}{6L^2} \quad \text{where } \begin{cases} \beta = +1 & \text{along the lines} \\ \beta = -\frac{1}{2} & \text{between the lines,} \end{cases}$$

$b$  is the correlation length parameter,  $r$  is the lag or separation,  $L$  is the number of Turning Band lines and  $K$  is a constant that depends on the covariance model. The error asymptotically decreases with the square of the number of lines, and increases linearly with the distance between the lines. As the size of the field increases, the spacing between lines increases, especially in areas farthest from the Turning Bands origin. *Mantoglou and Wilson* [1982] failed to realize the implications of this for large fields covering many correlation lengths. *Gutjahr* [1989] has shown that the errors have a periodic behavior. Because of the orientation of the errors observed by *Mantoglou and Wilson* [1982], along rays originating at the Turning Bands origin, the resulting field is in fact anisotropic for a finite number of lines. Spacing the lines non-uniformly on the unit circle does not eliminate the problem (*Mantoglou and Wilson* [1982], *Tompson et. al*, [1989]). With a conventional Turning Bands approach, the only solution appears to be increasing the number of Turning Band lines.

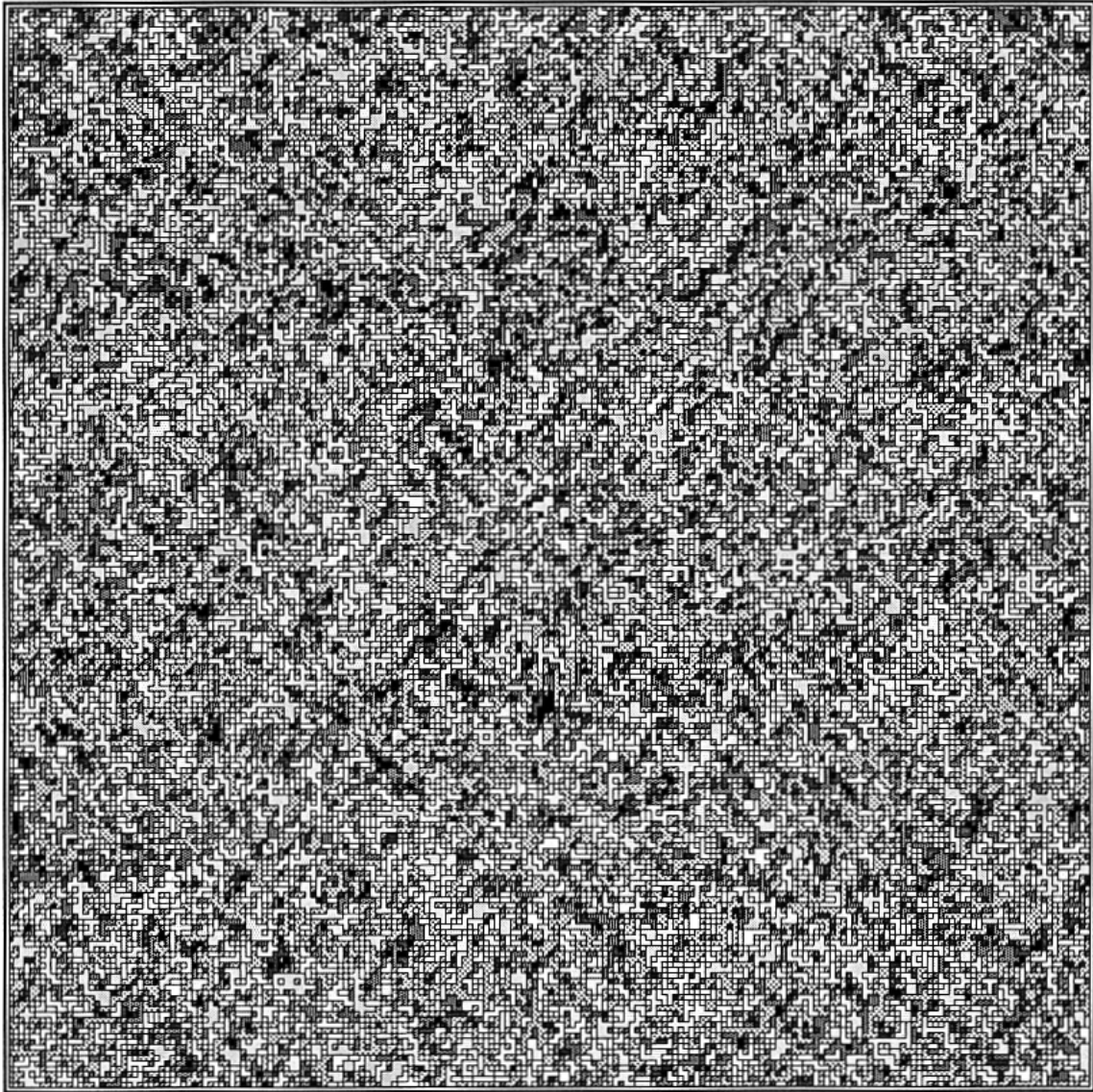


Figure 34. An exponential random field spanning 200 correlation lengths in each direction. Lineaments can be observed best by viewing the plot from several feet away and at an angle of approximately 20 degrees above the plane of the plot.

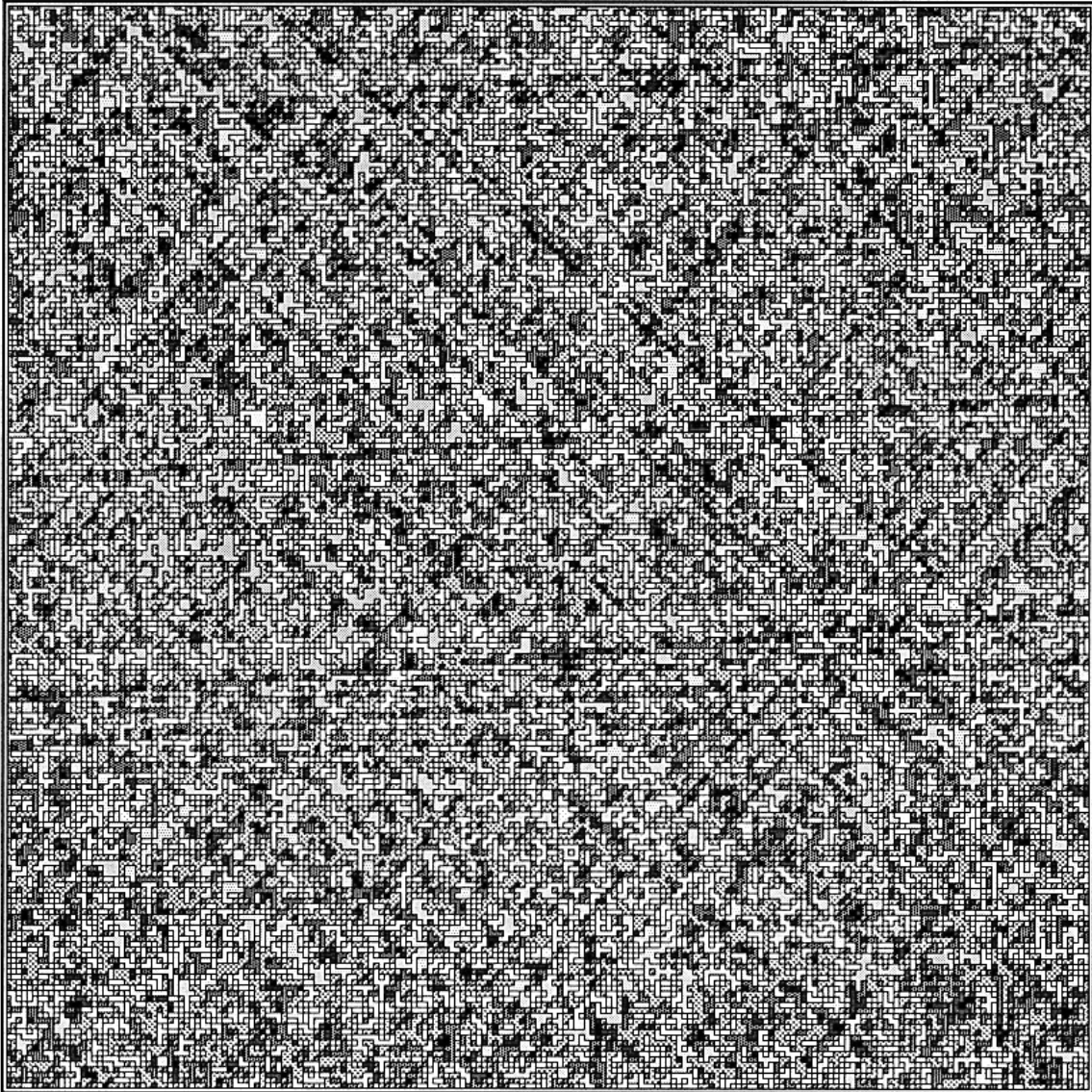


Figure 35. An exponential random field spanning 200 correlation lengths but generated using only 4 Turning Bands lines; the lineaments in this figure are much more noticeable.



TUBA version 2.10
-------------------

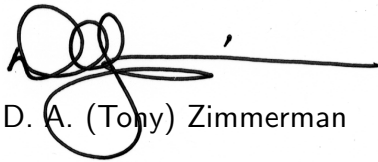
More “bells and whistles,” added conveniences, enhancements

With this new version of TUBA, there have been two major conveniences added and several enhancements to the code. The new capabilities included with this version are:

1. The ability to generate onto an irregularly-spaced finite difference grid (without specifying the coordinates of each nodal point)
2. The ability to reproduce the “ith” field from a multiple-simulation run without having to rerun all “n” fields (the random number generator seed is printed for each field).

In addition, several improvements in the coding and the I/O have been made and a *bug!* has been discovered and rectified. All of these changes to the code are described on the following pages along with some additional clarification of existing options in the code.

Regards,



D. A. (Tony) Zimmerman



## Code Modifications

In addition to writing new code to implement new options, the existing code was changed in several places to correct a bug, to make the code “smarter,” to improve I/O, and to eliminate some compiler-dependent problems.

### The “Bug” and its Fix

In version 2.0, a “floating divide by zero” will occur in the specific instance when one is attempting to generate an Intrinsic Random field (i.e., using the Generalized Covariance model) at arbitrary points in space with the default Turning Band parameters. Under these conditions, the variable **TBMX** (in subroutine **DEFPAR**) is used before it is set resulting in another variable, **UN**, being set to zero; **UN** is later used as a divisor in a quotient. This resulted from having the code choose a default value for **UN** (the Turning Band line discretization length) for this specific case. Normally, the guidance is to set **UN** to something smaller than the smallest grid spacing. However, there is no way to tell (without excessive computation) what the smallest spacing is between any two points when all the generation points are specified at arbitrary locations. The problem was corrected by changing the way the default value for **UN** is chosen. In version 2.10, an assumption is made concerning the geometry of the arbitrarily placed points; the assumption is that the points are uniformly distributed in space (as would be the case for an evenly-spaced grid). The fictitious grid spacing that results from this assumption is used to calculate a default value (0.2 times the grid spacing) for variable **UN** and eliminates the “divide by zero” problem.

This problem was discovered when TUBA was being used to generate unconditional realizations of the log-transmissivity field for the Culebra formation at the WIPP site in southeastern New Mexico. The user was attempting to generate random fields of log-transmissivity at the nodes of an irregularly-spaced finite difference grid consisting of over 5400 grid blocks. Obviously, to calculate distances between each node with every other node in order to find the minimum distance between any two generation points would require a significant computational effort (on the order of 60 million computations) for this large number of nodes.

### The Turning Bands Origin

The application described above also prompted an improvement in the manner of choosing the default Turning Bands origin for the case of generating at arbitrary locations in space. The computations will be minimized if the the Turning Bands origin is placed at the grid origin when generating onto a grid. In version 2.0, the default Turning Bands origin is always set to (0,0) with the assumption that this is also the *coordinate* origin. For generation at arbitrary points in space, however, this may not be the case. In the



WIPP site application described above, the generation points were described in state plane coordinates (whose origin is somewhere in the center of the state). Consequently, the lengths of the Turning Band lines were extraordinarily long, so long in fact, that the `I*4` variable holding the number of generation points along the line overflowed, causing the code to bomb. The temporary fix for this problem was to scale the coordinate values down (in effect bringing the coordinate origin closer to the field being generated). However, in version 2.10, this is no longer necessary. TUBA now places the default Turning Bands origin at the minimum-`X` and minimum-`Y` coordinate location.

### Avoiding Compiler Dependencies

Several code changes were made to make the code more resistant to compiler dependencies. For example, in subroutine `FILPAR`, the manner of calculating an internal parameter, `IOF`, was changed. In TUBA, there is an implicit assumption that uninitialized variables are set to zero; this will not always be the case, depending on the compiler being used. It appears that, with the exception of this case for variable `IOF`, this assumption is inconsequential, as all variables are set prior to being used. Another implicit assumption in the coding is that values of all local variables are saved after leaving a subroutine (there are no `SAVE` statements in the code). This does not appear to cause any problems even for compilers which require explicit `SAVE` statements, probably because all variables are either in `COMMON` or are passed as subroutine arguments.

One problem which occurs when compiling TUBA on certain compilers, is the compiler's treatment of incompatible type declarations between the calling program and the subroutine as an error rather than a warning. TUBA uses memory very efficiently and does so by storing all array data in a single vector of type `REAL`. In subroutine `SPCTRL`, TUBA version 2.0 passes an address (position within the array) of the real array to subroutine `FFTGEN` whereupon the array is declared to be of type `COMPLEX`; this causes some compilers to treat this condition as an error. This will not cause any errors (in output) provided the memory management within the code is done correctly (which it is). The compiler problem was solved by making the single array used for array storage both real *and* complex via an equivalence statement in the main program module.

### Improvements in I/O

The only significant changes that were made in I/O were made in subroutine `RDINTG` where `F` formats were changed to `G` formats and the length of filenames were increased to 35 characters. The `G` format will represent very large or very small numbers in scientific notation (`E` format) when necessary, whereas the `F` format will not. In applications, the `F` format represented very small numbers (e.g., Generalized Covariance model coefficients) as zeros in the list file, thus misrepresenting the input to the code.

## Miscellaneous Notes

Comments in the source code have been improved slightly. Don't forget about the ability to run TUBA "in batch mode" via redirection symbols, it will save much typing and time (see Section 4.2). There has been some confusion concerning the sign on the Generalized Covariance (GC) model coefficients, **A1**, **A3**, **A5**. TUBA has been coded with a GC model formulation which is consistent with that described in *Delfiner*, [1976]. Use the guidance the code requires which is provided during execution.

## Reproducing the "ith" Simulation Using Two RNG Seeds

TUBA includes the option of generating multiple random fields for a given set of covariance model and field geometry specifications. Typically, this option would be invoked for performing Monte Carlo simulations of some process using these fields as input. On the other hand, one may be searching for a particular type of 'character' in the field which requires searching through many realizations of the field until a satisfactory one is found. In this case, it would be convenient to be able to reproduce that one field without having to generate the entire suite of fields that lie ahead of it. TUBA version 2.10 allows you to do this by specifying two random number generator seeds instead of one. The first seed must be the same as the seed given for the multiple simulation run; the second seed is the seed which was generated internally and printed in the list (.INP) file just above the summary statistics for each simulation.

Shown below is the input data file for a case where five simulations of a random field with an exponential covariance structure were generated. Following that is output which shows summary statistics for each of the five fields. Recall that the output list file has the same name as the output data file(s) save for the extension .INP.

### Input Data for Multiple Simulation Run [MEXPO.INP]

	2		1=(x,y) Locations, 2=Gridded output
	2		1=Point Centered, 2=Block Centered
24.000	48.000		Maximum X and Y Field Dimensions
	24	48	Number of Nodes-X and Nodes-Y
none			Mask Filename
	1		1=Normal, 2=exp(X), 3=10**(X)
	1		0=User,1=Exp,2=Gauss,3=Bes1,4=Telis,5=GC
	1		1=Point Process, 2=Areal Average Process
0.00000	1.0000		Desired Mean and Variance
12.000	12.000		X and Y Direction Correlation Lengths
MEXPO.DAT	1		1=Default TBM Parameters, 2=Enter Manually
			Output Data Filename
(16F8.4)	2		1=Unformatted, 2=Formatted Output
			Output Format for Writing Data to Disk
	2		1=Single Write Statement, 2=Line at a Time
	1		1=First Row to Last, 2=Last Row to First
	2		1=Marsaglia URNG, 2=Machine Indep URNG
12345678			Seed(s) for Random Number Generator
	5		Number of Realizations to be Simulated

Output Data from Multiple Simulation Run [MEXP5.INP]

Field ORIGIN relative to TBM origin = 0.000000 0.000000  
 Number of Turning Band Lines Equals = 16  
 The Maximum Turning Band Line Length = 53.6656  
 Turning Band Line Discretization Lgth = 0.750000  
 Maximum Frequency for the Spectrum = 100.531  
 Number of Harmonics for the Spectrum = 2048  
 Frequency Spacing in Spectral Domain = 0.490874E-01  
 Spatial Discretizations, DELX & DELY = 1.00000 1.00000  
 No Pnts/correlation Length in X,Y Dir = 12.0 12.0  
 Approx Number of Independent Samples = 2.0

Output Filename = MEXPO.1  
 New Random Seed = 12345678  
 The Sample Mean = 0.5418  
 Sample Variance = 1.1183

Output Filename = MEXPO.2  
 New Random Seed = 50882505  
 The Sample Mean = 0.9901  
 Sample Variance = 0.8302

Output Filename = MEXPO.3  
 New Random Seed = 25499530  
 The Sample Mean = 0.4384  
 Sample Variance = 0.5672

Output Filename = MEXPO.4  
 New Random Seed = 38541230  
 The Sample Mean = 0.1433  
 Sample Variance = 0.2950

Output Filename = MEXPO.5  
 New Random Seed = 30806103  
 The Sample Mean = -0.0902  
 Sample Variance = 0.9034

The Ensemble Mean = 0.40467  
 Ensemble Variance = 0.87752

Suppose we wished to reproduce the last field shown above, MEXPO.5 because its sample statistics are close to the theoretical values. This can be accomplished by copying the file MEXPO.INP to MEXP5.INP and making three minor changes. Adding the random seed value shown in the output above, changing the number of simulations from 5 to 1, and changing the name of the output file. The modified input file is shown below. A file comparison of MEXPO.5 against MEXP5.DAT showed no differences.

Input Data for Two Random Seeds Problem [MEXP5.INP]

	2		1=(x,y) Locations, 2=Even Grid, 3=Uneven
	2		1=Point Centered, 2=Block Centered
24.000		48.000	Maximum X and Y Field Dimensions
none	24	48	Number of Nodes-X and Nodes-Y
			Mask Filename
	1		1=Normal, 2=exp(X), 3=10**(X)
	1		0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC
	1		1=Point Process, 2=Areal Average Process
0.00000		1.0000	Desired Mean and Variance
12.000		12.000	X and Y Direction Correlation Lengths
MEXP5.dat	1		1=Default TBM Parameters, 2=Enter Manually
			Output Data Filename ← changed
(16F8.4)	2		1=Unformatted, 2=Formatted Output
			Output Format for Writing Data to Disk
	2		1=Single Write Statement, 2=Line at a Time
	1		1=First Row to Last, 2=Last Row to First
	2		1=Marsaglia URNG, 2=Machine Indep URNG
12345678 30806103			Seed(s) for Random Number Generator ← two seeds
	1		Number of Realizations to be Simulated ← changed

## Generating onto an Irregularly-Spaced Finite Difference Grid

In this example, we generate a random field onto a small (7 x 6) point-centered but *irregularly spaced* finite difference grid. The field covariance properties are exactly the same as the field generated in Section 4.2.5. The nodes which define the irregularly-spaced finite difference grid (Figure 38) are coincident with the some of the nodes of the regularly-spaced finite difference grid shown in Figure 21 (where, in an analogous fashion, a finite element grid is superimposed). The irregularly-spaced finite difference grid used for this example has only 42 nodes (the boxed values in Figure 38) and it would be relatively painless to specify the coordinates of each node and generate the random field using the procedure described in Section 4.2.5. However, for very large grids (such as the grid used for the WIPP site application described above where there were 5400 nodes), an easier alternative is available.

With TUBA version 2.10, an abbreviated description of the grid spacing is read from a file and generation of the field at the nodes of the irregularly-spaced finite difference grid is taken care of automatically. The example input files and output are shown below.

### Input File of Grid-block Widths [ IRSG.GBW ]

```
2 1 1 1 2 3
3 2 1 1 1
```

### Input Data for Irregularly-Spaced Grid [ IRSG.INP ]

	3			1=(x,y) Locations, 2=Even Grid, 3=Uneven	← option 3
	1			1=Point Centered, 2=Block Centered	
	10.000	8.0000		Maximum X and Y Field Dimensions	
	7		6	Number of Nodes-X and Nodes-Y	
IRSG.GBW				Input Filename for grid-block widths	← new
NONE				Mask Filename	
	1			1=Normal, 2=exp(X), 3=10**(X)	
	4			0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC	
	0.00000	1.0000		Desired Mean and Variance	
	5.0000	5.0000		X and Y Direction Correlation Lengths	
			2	1=Default TBM Parameters, 2=Enter Manually	
			16	Number of Turning Band Lines	
	0.31250			TBM Line Discretization Distance	
	0.25000			Discretization Distance for MA Process	
	0.00000	0.00000		Field ORIGIN Relative to TBM Origin	
	12.806			Maximum Turning Band Line Length	
IRSG.DAT				Output Data Filename	
	2			1=Unformatted, 2=Formatted Output	
(7F8.3)				Output Format for Writing Data to Disk	
	2			1=Single Write Statement, 2=Line at a Time	
	2			1=First Row to Last, 2=Last Row to First	← note
	2			1=Marsaglia URNG, 2=Machine Indep URNG	
52379164				Seed(s) for Random Number Generator	
	1			Number of Realizations to be Simulated	

### Output Data for Irregularly-Spaced Grid [ IRSG.DAT ]

```
3.303 1.772 2.508 2.091 2.435 0.760 0.130
3.046 1.807 2.838 2.046 2.811 1.471 0.423
2.239 3.421 2.967 2.597 3.109 1.527 0.647
2.110 3.032 3.369 3.055 2.694 1.713 1.356
0.198 2.056 1.271 2.207 1.193 0.166 1.799
0.555 0.358 0.516 0.450 1.418 1.361 0.328
```

Thus, with TUBA version 2.10, there is a new option for the very first question asked by the code; you are now prompted with the following:

```

+++++++ OUTPUT FIELD PARAMETERS ++++++
(1) - Simulate Only At Specified (x,y) Locations
(2) - Simulate Onto A Regularly Spaced Grid
(3) - Simulate Onto An Unevenly Spaced Grid

```

When you respond with “3” for the third option above, TUBA will ask for the name of the file that contains the “grid-block widths”. The file used in this example is named `IRSG.GBW` and contains only two lines; the first describes the widths of the grid blocks in the X-direction while the second describes the grid-block widths along the Y-direction. In Figure 36 (as well as Figure 21), the 0--10 and 0--8 values on the abscissa and ordinate respectively, are the *coordinate values* corresponding to the regularly-spaced grid nodes. Hence, the grid-block widths for this example can be determined by inspection of Figure 36. Note that the data values listed in `IRSG.DAT` match exactly the boxed values shown in Figure 36.

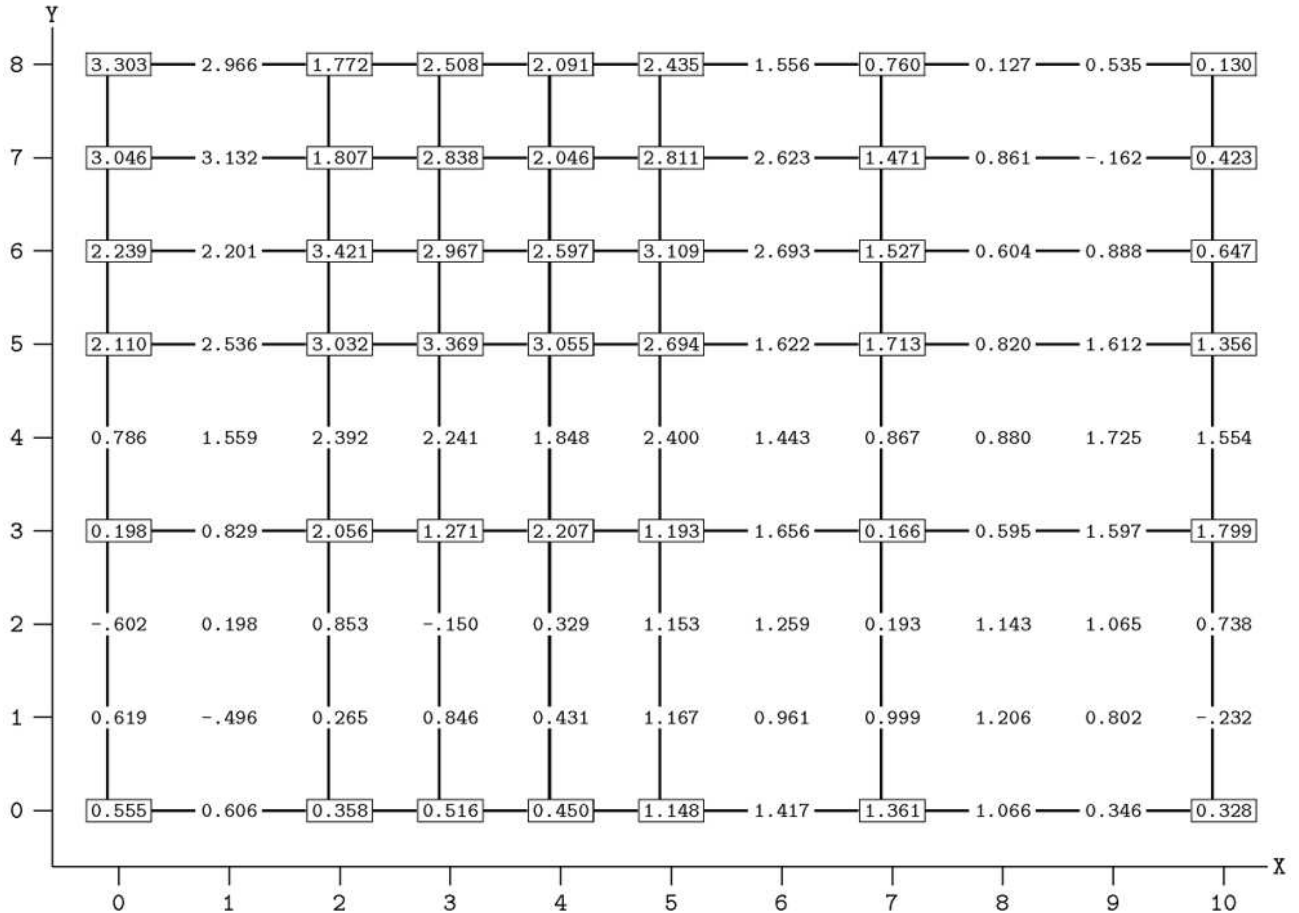


Figure 36. A point-centered, irregularly-spaced finite difference grid.

## The Ergodicity of the Turning Bands Algorithm

The ergodicity of the Turning Bands algorithm was demonstrated in part, in Section 4.3.1. In this example, the “other half” of that demonstration is carried out. Here, we use the exact same input parameters as was used for the exponential field in Section 4.2.1, but generate 100 realizations instead of just one. When more than one simulation is being generated, TUBA writes the mean and variance statistics for the ensemble of fields at the end of the file. As shown below, the ensemble statistics converge to the theoretical values. The input and (some of the) output data is listed below.

```

          2          1=(x,y) Locations, 2=Even Grid, 3=Uneven
          2          1=Point Centered, 2=Block Centered
    100.00    100.00    Maximum X and Y Field Dimensions
          100    100    Number of Nodes-X and Nodes-Y
NONE          Mask Filename
          1          1=Normal, 2=exp(X), 3=10**(X)
          1          0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC
          1          1=Point Process, 2=Areal Average Process
    0.000E+00 0.000E+00 1.000E+00 Desired Mean, Nugget and Sill
    20.000    20.000    X and Y Direction Correlation Lengths
          16    Number of Turning Band Lines
          1          1=Default TBM Parameters, 2=Enter Manually
ERGO.DAT          Output Data Filename
          1          1=Unformatted, 2=Formatted Output
          2          1=Single Write Statement, 2=Line at a Time
          2          1=Marsaglia URNG, 2=Machine Indep URNG
57756341          Seed(s) for Random Number Generator
          100    Number of Realizations to be Simulated
          1          1=Single file output, 2=Multiple files
          0          0=do not scale, 1=match T-stats exactly
          2          1=Minimal, 2=Med, 3=Frequent screen output

```

(reflection of TUBA’s internal Turning Bands parameters suppressed)

```

Simulation Nmbr =      1
Output Filename = ERGO.DAT
New Random Seed =  57756341
The Sample Mean = -0.13424
Sample Variance =  0.80305

Simulation Nmbr =      2
Output Filename = ERGO.DAT
New Random Seed =  10507430
The Sample Mean =  0.71254
Sample Variance =  0.70058

```

(statistics for fields ERGO.003 through ERGO.098 deleted from the list)

```

Simulation Nmbr =      99
Output Filename = ERGO.DAT
New Random Seed =  77434250
The Sample Mean = -0.0090872
Sample Variance =  0.78407

Simulation Nmbr =     100
Output Filename = ERGO.DAT
New Random Seed =  35774455
The Sample Mean =  0.60989E-01
Sample Variance =  1.1500

```

```

THE ENSEMBLE STATISTICS ...
The Ensemble Mean = -0.036318
Ensemble Variance =  0.98782

```

← Note convergence of  
the ensemble statistics

TUBA version 2.11
-------------------

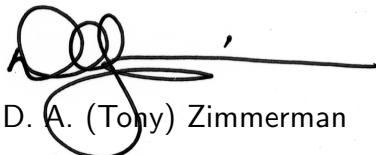
## “A Much More Memory Efficient TUBA”

With this new version of TUBA, the code structure has been modified to reduce the amount of storage (memory) required to perform the calculations, to permit the generation of very large data sets and to enable a large number of Turning Band lines to be used. The changes which are visible to the user include the following new options:

- 1 For specification of the Turning Band parameters, the user is now first prompted for the number of Turning Band lines and secondly prompted for the option to, as before, either accept the remaining default Turning Band parameters or specify them manually.
- 2 For specification of the covariance model parameters, the user now has the option to specify the desired mean and variance for IRF-k fields generated with the Generalized Covariance model.
- 3 When simulating multiple fields, the output fields can be written to multiple files as before (e.g., with filenames such as outname.1, outname.2 etc.) or the data for all the fields can be written into a single file (e.g., outname.dat).
- 4 An option to specify the level of “progress reporting” during the calculations has been added; three levels are provided, 1=minimal, 2=more frequent and 3=very frequent status reporting.

Most of the changes incorporated into version 2.11 were made to enable the generation of very large fields (many correlation lengths), very dense data sets and the use of many Turning Band lines. For example, a field of size 1000 × 1000 (one million nodes) was generated using 256 Turning Band lines – the required dimension of the A array was only 1,018,900 (using a Gaussian covariance function) and the execution time was only 3 hours, 22 minutes on a 486/33 PC! (37 seconds on a 2.2 GHz AMD Athlon XP 3200)

Regards,



D. A. (Tony) Zimmerman

## **Memory Requirements Versus Execution Speed**

The manner in which the Turning Bands method calculations are performed in this version of TUBA involves unformatted direct access Fortran I/O which is the fastest type of disk access available with Fortran. The generation points (coordinates) of the output field are saved in a scratch direct access file as this information is accessed repeatedly (once for each Turning Band line) during execution of the program (A scratch file is deleted upon termination of the program). In previous versions of TUBA, this information was kept in memory drastically increasing the amount of memory required to perform the simulations. In addition, when the areal average process is chosen (currently programmed only for the exponential covariance model), initialization of the line process is different for each Turning Band line. Therefore, when areal averaging is used, the line process calculations are also stored in an unformatted scratch direct-access file.

Because disk I/O is used to store and retrieve this information instead of memory, this new version of TUBA will be somewhat slower than previous versions. However, for small fields (say 100 x 100 or less) the overall execution time is so small that the extra increment of time is inconsequential. For large fields or fields generated with many Turning Band lines, the amount of memory required under the original computation scheme would be prohibitive. Hence, this new version of TUBA represents a substantial improvement in the versatility of the code to simulate random fields.

When running TUBA on a PC under DOS, the slight decrease in execution speed can be avoided by using a RAMDRIVE and running TUBA on that drive. A RAMDRIVE is a section of memory that can behaves like a disk drive, i.e., when the data are written to the scratch file, it is actually written to memory rather than disk. A 20 to 25 percent increase in speed was observed for fields ranging in size from 125,000 to 500,000 nodes when TUBA was run from the RAMDRIVE. This approach can also be implemented on some unix systems.

## **Specifying Generalized Covariance Model Parameters**

When generating Intrinsic Random Fields of Order-k with the Generalized Covariance (GC) models, the output is now scaled to specified mean and variance values. In earlier versions, the generated IRF-k values would start at an arbitrary “base level” (zero) and yield a field with a very high variance.

The formula on page 11 (the one labeled “DO NOT USE THIS”) is used to scale the generated field values to match the target mean and variance; equation (3) on page 12 can not be used since there is no “original theoretical mean and variance” values specified for GC models.



Thus, for the GC models, the interactive input now looks like the following:

```

+++++++ COVARIANCE PARAMETERS ++++++
Select Type Of Covariance Model:
(0) - User Specified
(1) - Exponential Model
(2) - Gaussian Covariance
(3) - Bessel Type Covariance
(4) - Telis Covariance Function
(5) - Generalized Covariance Model
>>>> 5

Enter Gen. Covariance Parameters A1,A3,A5
      K(r) = A1*r + A3*r**3 + A5*r**5
{ A1,A5.GE.0, A3.GE.-(10/3)*SQRT(A1*A5) }
>>>> 0.000000 1.00000 0.000000

Enter The Desired Mean And Variance
>>>> 0.000000 1.00000

```

### **Specifying Turning Bands Parameters**

As the size of the field (i.e., number of correlation lengths) becomes larger, the number of Turning Band lines needs to be increased. Although there is no theoretical or empirical formula for determining precisely how many lines should be used, there has been some concern raised among geostatisticians (see Section 4.3.6 and *Tompson et al.*, [1989]) with regard to the presence of “lineaments” appearing in the field when an insufficient number of lines are used. Therefore, the user must now first prescribe the number of Turning Band lines to be used, and then choose to either accept the default TBM parameter values or specify them manually as follows:

```

+++++++ TURNING BANDS PARAMETERS ++++++

Enter The Number Of Turning Band Lines
      { Use At Least 16 }
>>>> 16

For The Remaining Turning Band Parameters:
(1) - Use Default Turning Band Parameters
(2) - Enter The TBM Parameters Manually
>>>> 1

```

### **New Simulation Parameters Options**

The other new options arise during specification of the simulation parameters. In the example below, all fifty realizations are written to a single file instead of fifty separate files. This may be more convenient for postprocessing the data with other programs.

Several options are now available for reporting computation progress to the screen. This provides a way to gage the computational burden and status of a long run. For example, one may wish to specify minimal reporting for a large multiple simulation run, more frequent reporting for a field with many Turning Band lines, and very frequent reporting for very large fields.

Thus, the input stream for the simulation parameters now appears as follows:

```
+++++ SIMULATION PARAMETERS +++++
(1) - Marsaglia and Bray Random Number Generator
(2) - Machine Independent Random Number Generator
>>>> 2

Enter Integer Seed(s) To Initialize The Generator
{ Seed For This Generator Must Be 8 Digits Long }
>>>> 12345678

Enter The Number Of Realizations To Be Simulated
>>>> 50

(1) - All realizations written to one file
(2) - A separate file for each realization
      (e.g., file.1, file.2 ... file.99)
>>>> 1

Specify The Level Of Status Reporting To The Screen
(1) - minimal (e.g., for many realizations)
(2) - more frequent (e.g., for many TBM lines)
(3) - very frequent (e.g., for very large fields)
>>>> 2

Number Of Elements Allocated In A Array = 50000
Total Storage Required For Computations = 9425
No Of Elements In Excess of Requirements = 40575
Free Disk Space Needed For Computations = 9.600 Kilobytes
```

Note also that the A-array storage requirements and the amount of free disk space required are also listed to the screen and the bottom of the <NAME>.INP file.

### **Effect of Change in Default Turning Bands Line Discretization Length**

In version 2.11, a change was made in subroutine DEFPAR, the subroutine which computes default Turning Band parameter values. The default discretization length for the white noise process of the Moving Average algorithm (parameter DS, Telis covariance model) was changed in accordance with the guidance provided in subroutine TBMPAR.

*Mantoglou and Wilson*, [1981] point out that the discrete approximation of the covariance function involves some error which increases as the discretization length DS increases. The default DS is now computed so that it is no larger than 1/10th the Turning Band line discretization interval. As a result of this change, the mean and variance statistics of Telis fields shown in the manual (pages 45, 66, 67, 74 and 89) will not match those output by TUBA version 2.11 nor will the fields look the same. However, the examples in the manual are intended only for illustrative purposes and for instruction on how to properly apply the code.

## Chapter 5

### Programmer's Manual

This chapter is intended to provide a brief description of the TUBA computer code that will enable computer programmers to decipher the algorithms and make whatever changes may be necessary to accommodate the local computing environment. Code segments which may not be standard on every computer operating system are discussed and certain operational features of the code are explained.

TUBA was originally written in Fortran IV for batch processing on an IBM computer. During the 1988 – 1990 revision process, the code was completely reorganized, modified to take advantage of Fortran 77 programming features, and written in a style amenable for both batch or interactive processing. TUBA should, without modification, or with possibly only minor modifications, compile and run on almost any computer operating system; it has been tested under VMS, Solaris, HPUX, Linux, Macs and PC's running Windows (all flavors).

#### § 5.1 Redimensioning

A problem commonly encountered when implementing a new code is figuring out how to properly change the dimension statements for your particular application. With TUBA this is not a problem; there is only one array in TUBA, and the code tells you during execution the required dimension size for the problem you are running. The first dimension statement in the main program declares the variable **A** as an array of length **LGTH** where **LGTH** is defined as a parameter set to some arbitrarily large value, say, **LGTH = 999000**. If the allocated storage space is insufficient, program execution halts and a message is issued instructing the user to recompile the program with **LGTH** set to the required size. To run TUBA in the DOS operating environment, **LGTH** must be set to a much smaller value (say  $\text{LGTH} \approx 116000$ ) because of DOS memory constraints †.

Individual arrays are accessed through “array pointers” which are integer variables that indicate the starting location of each array within the **A** array. This technique enables all data to be efficiently stored and retrieved and it allows the generation of fields of varying grid sizes without having to adjust the array dimensions. The addresses (array pointer values) of the arrays are calculated in subroutine **ADDRES**.

---

† More recent Fortran compilers for DOS are not limited to 640K of RAM.

## § 5.2 Code Structure and Programming Notes

TUBA has been modularized almost to the extreme – the reason for coding it this way is simply style; it is the author’s belief that each module or subroutine should perform a single task which is easily identifiable by examining the code. Usually, a one line comment just under the subroutine declaration is sufficient to describe the purpose of the module; an understanding of how the module’s objective is accomplished can be obtained by examining the code. Because each module is short, there should be little difficulty developing an understanding of what each module’s function is and how the task is performed. The order in which the various computational tasks are performed in TUBA is shown in Figure 37. Routines not shown include RDCHAR, RDINTG, RDREAL, NCHR, COMENT, PROGSS, URN55, URNSS and URNMB; these are general purpose utility modules whose function can be determined by examining the respective code listings. Subroutines in the code listing are arranged alphabetically.

TUBA is written in a style that is intended to facilitate one’s comprehension of the computational tasks. Nevertheless, there may be some programming techniques used that appear to be somewhat obscure. Understanding how some of these techniques are used will better enable one to follow the code and/or make code modifications if desired. The manner in which arrays are passed to subroutines was discussed in section 5.1. Subroutines RDINTG and RDREAL are used to read and reflect integer and real input data respectively. These routines read the input data into an integer or real array which is passed in the argument list of the subroutines (RDREAL is actually an ENTRY statement in subroutine RDINTG). The calling program, however, doesn’t pass an array, rather, it passes a scalar variable which is in common with the other variables that the program is attempting to read. For example, NX and NY (the nodal field dimensions for a gridded output field) are placed next to each other in the labeled common statement, /TBMPAR/, and are read at the same time even though only NX is passed in the call to subroutine RDINTG (this particular call is made in subroutine FLDPAR).

The manner in which the “card file” <NAME>.INP is created can be determined by examining the LSTINP subroutine. TUBA creates this card file as it is being run interactively; the file can then be used later for batch processing, or interactive processing with redirected input. As input parameters are read (via subroutines RDINTG, RDREAL, RDCHAR), they are written to a scratch file and annotated (subroutine COMENT provides the annotations); after all the input data are read and the internal parameters are calculated, the input data and their annotations are reread from the scratch file and written out along with the internal parameters to the <NAME>.INP file.

```

MAIN    - main program; driver which calls all of TUBA's subroutines
|
RDINPT - control module for reading input parameters
|   FLDPAR - read output field parameters
|   COVPAR - read covariance model parameters
|   TBMPAR - read Turning Band parameters
|   |   DEFPAR - calculate default Turning Band parameters
|   |   |   ORGMAX - calculate TB origin and max TB line length
|   SIMPAR - read simulation parameters
|   |   UNITMB - initialize Marsaglia and Bray random number generator
|   |   UNITSS - initialize Swain and Swain random number generator
INTPAR - calculate internal Turning Band and covariance parameters
LSTINP - list the input data and some internal parameters
ADDRES - calculate array addresses (array pointer values)
CALXYP - calculate (x,y) projection point parameters
|   CALXYC - calculate (x,y) coordinates for gridded output
CALINP - calculate line process array data
|   SPDF   - spectral density function for various covariance models
|   WTEXP - calculate weights for exponential areal average process
|   WTUSR - calculate weights for user-defined areal average process
|
BEGIN DO LOOP - repeat this loop for each simulation
|   SPCTRL - spectral generation of line process
|   |   FFTGEN - driver for FFT line process generation algorithm
|   |   FFT    - Fast Fourier Transform algorithm
|   MOVAVG - moving average generation of the line process
|   WNRLVY - Weiner-Levy generation of the line process
|   PROJCT - project line process data onto the output field
|   |   PROJSB - utility subroutine called by PROJCT
|   OUTPUT - write field(s) to the output file(s)
|   |   FSCALE - final scaling and mean and variance calculations
|   |   OPNFIL - open output file and list random field statistics
END DO LOOP

```

Figure 37. Order of subroutine calls in TUBA.

## Miscellaneous Programming Notes

Some compilers may issue a warning when variable types of subroutine arguments are mismatched between the calling program and the subroutine itself. For example, subroutine SPCTRL passes a real array, DZ, to subroutine FFTGEN; in FFTGEN, the DZ array is declared complex. Any non-fatal compiler error or warning message concerning this may be ignored.

When unformatted output is requested, the open statement for the output data file contains a record-length specifier (LREC). The variable LREC is the maximum required record-length size (in bytes); LREC is calculated in subroutine OUTPUT while the open statement is located in subroutine OPNFIL. Depending on the compiler being used, the record-length specifier may be omitted and may, if not omitted, cause a compiler error. Also, on some compilers, the record-length specifier may refer to the number of words in the record rather than the number of bytes.

### § 5.3 Random Number Generators and the Fast Fourier Transform

In many instances, codes which require special mathematical or statistical calculations rely on “packaged software” such as IMSL (International Math and Science Library) or SAS (Statistical Analysis Software). TUBA is a “self-contained code” in that no additional libraries or subroutines need to be linked in order to make it run on any computer system. The source code for special mathematical or statistical computations is included in the program listing. Two of these, random number generation and the discrete Fast Fourier Transform (FFT), are discussed below.

Two random number generators are included in the TUBA code, one is machine dependent, the other machine independent. References for each are listed in the source code; both have been tested extensively with the TESTRAND code (*Dudewicz and Rally* [1981]), and both were modified slightly from their original form. The modifications were made so that both the generator initialization and the random number generation are contained in one module and so that each generator returns uniformly distributed random numbers on the interval  $[-0.5, +0.5]$  rather than on  $[0, 1]$ .

The machine dependent generator (algorithm of Marsaglia and Bray presented on pages 567 and 597 of *Dudewicz and Rally* [1981]) was “recommended for practical use” by the above authors; it passed very sensitive and exhaustive testing. This generator relies on integer-overflow arithmetic where the high-order bits of an integer product are ignored if the number of bits required to represent the product is greater than the number of bits in the integer word size. On some compilers, this action may result in a run-time error unless the code is compiled with the “integer overflow check” compiler switch turned off.

The machine independent generator (*Swain and Swain* [1980]) failed the rigorous Chi-square on Chi-square test performed by the TESTRAND code. However, this test is an extremely sensitive test and failure does not preclude its use for the generation of random fields. A plot of  $X_j$  versus  $X_i$  where  $j = i + 1$  and the  $X_i$  are random numbers produced with this generator is shown in Figure 37. This plot represents a kind of graphical test of whether the deviates are sequentially correlated; the plot shows a uniform scattering of points with no “holes” indicating that the deviates are uncorrelated and uniformly distributed. This generator uses integer arithmetic (without overflow) such that random number sequences can be reproduced exactly on different machines with different compilers and different word sizes. All the random fields generated in this manual were generated using the Swain and Swain random number generator.

The discrete Fast Fourier Transform (FFT) algorithm is used for calculating the spectrum of the line processes for some of the stationary covariance models. The algorithm used in TUBA was taken from *Gonzales* [1987]; the routine was modified slightly to enable it to calculate both forward and inverse transforms. Note that the transform algorithm is only valid for arrays of length equal to  $2^n$  for some  $n$ . The routine was verified by comparing its results with those obtained using FFT routines from the IMSL library.

## § 5.4 User-defined Covariance Models

Covariance models other than those described in this manual can be used to generate stationary random fields having either point or areal average properties; the line process for the user-defined covariance model can be generated using either a spectral method or a moving average method. The one-dimensional spectrum of the user-defined covariance model must be coded and inserted into TUBA; additional coding is required for implementing the moving average method or the areal average process.

Code for the one-dimensional spectrum must be inserted into function SPDF (see annotations in the code listing). If areal averaging is to be used, code for calculating the areal average weights (see section 2.6) must be inserted into function WTUSR. If a user-defined moving average method is used, the moving average weights (see section 2.4) must be calculated in subroutine CALINP and stored in the FF array. Also, variable CLN in subroutine INTPAR must be set equal to some value representing the number of correlations lengths the moving average weighting function is substantially non-zero.

Random Number Generator Plot Test  
 $X(i)$  vs  $X(j)$  where  $j = i+1$   
URN Function, NPTS = 1000

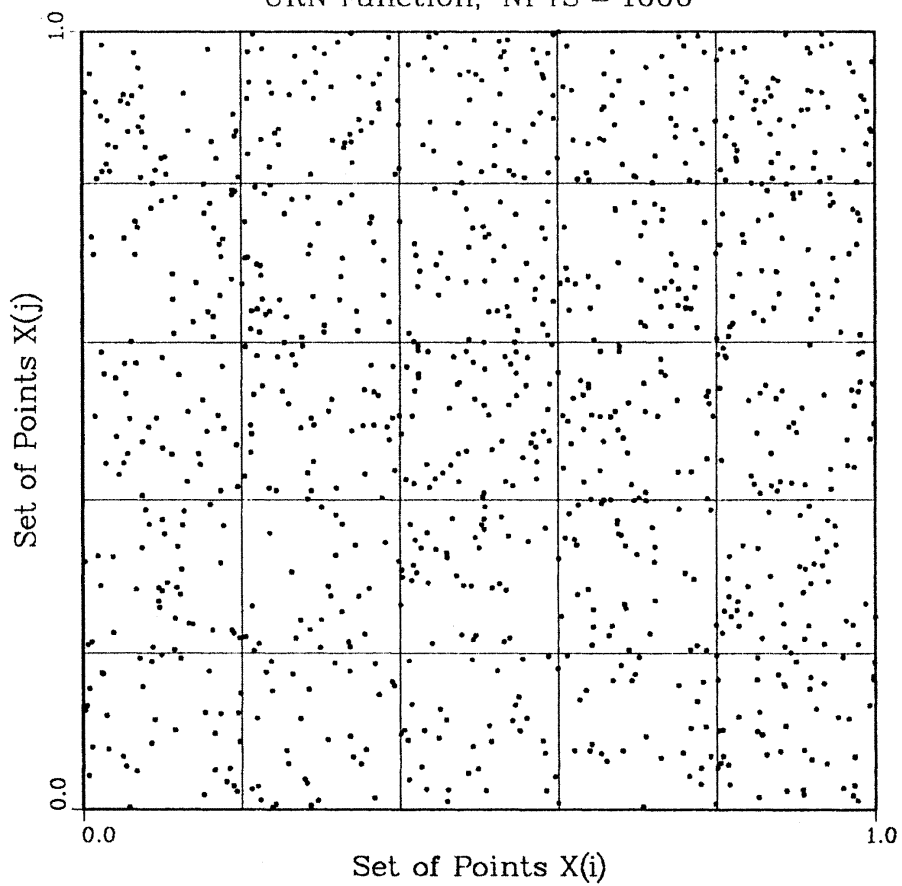


Figure 38. Graphical “test” of random number generator showing a uniform coverage of the data interval.



## References

- Abramowitz, M. and I. A. Stegun, 1964. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, National Bureau of Standards Applied Mathematics Series · 55, U.S. Government Printing Office, Wash. D.C.
- Bakr, A. A., Gelhar, L. W., Gutjahr, A. L. and J. R. MacMillan, 1978. Stochastic analysis of spatial variability in subsurface flows, 1. Comparison of one- and three-dimensional flows, *Water Resour. Res.*, *14*(2), 263–271.
- Box, G. E. P. and G. M. Jenkins, 1970. *Time Series Analysis, Forecasting, and Control*, Holden-Day, Inc., San Francisco, CA.
- Bras, R. L. and I. Rodriguez-Iturbe, 1984. *Random Functions and Hydrology*, Addison-Wesley Publishing Co., Reading, MA.
- Delfiner, P., 1976. Linear estimation of nonstationary spatial phenomena, In *Advanced Geostatistics in the Mining Industry*, edited by M. Guarascio, M. David and CH. J. Huijbregts, D. Reidel Pub. Co., Hingham, MA.
- Delfiner, P., 1978. “The intrinsic model of order k”, summer school notes, Centre de Géostatistique, Fontainebleau, France.
- Delhomme, J. P., 1979. Spatial variability and uncertainty in groundwater flow parameters: A geostatistical approach, *Water Resour. Res.*, *15*(2) 269–280.
- Dudewicz, E. J. and T. G. Ralley, 1981. *The Handbook of Random Number Generation and Testing with the TESTRAND Computer Code*, American Sciences Press, Inc.
- Freyberg, D. L. and T. C. Black, 1987. “Simulation of one-dimensional correlated fields using a matrix-factorization moving average approach”, Dept. Civ. Eng., Stanford University, Stanford, CA.
- Frind, E. O., Sudicky, E. A. and S. Schellenberg, 1987. Micro-scale modelling in the study of plume evolution in heterogeneous media, *Stochastic Hydrology and Hydraulics*, *1*, 263–279.
- Gelhar, L. W., 1984. Stochastic analysis of flow in heterogeneous porous media, In *Fundamentals of Transport in Porous Media*, J. Bear and M. Corapcioglu, eds., Martinus Nijhof, Dordrecht, Netherlands.
- Gomez-Hernandez, J. J. and S. M. Gorelick, 1989. Effective groundwater model parameter values: Influence of spatial variability of hydraulic conductivity, leakage, and recharge, *Water Resour. Res.*, *25*(3), 405–429.

- Gonzales, R. C., 1987. *Digital Image Processing*, Addison-Wesley Publishing Co., Reading, MA.
- Gutjahr, A. L., 1989. "Fast fourier transforms for random field generation, Project report for Los Alamos Grant to New Mexico Tech", Dept. Mathematics, New Mexico Institute of Mining and Technology, Socorro, NM.
- Jenkins, G. M. and D. G. Watts, 1968. *Spectral Analysis and its Applications*, Holden-Day, Inc., Oakland, CA.
- Journel, A. G., 1974. Geostatistics for conditional simulation of ore bodies, *Econ. Geol.*, *69*, 673–687.
- Journel, A. G. and CH. J. Huijbregts, 1978. *Mining Geostatistics*, Academic Press, New York, NY.
- Kafritsas, J. and R. L. Bras, 1981. The practice of kriging, *Tech. Rep. No. 263*, Dept. Civ. Eng., Massachusetts Institute of Technology, Cambridge, MA.
- Kafritsas, J. and R. L. Bras, 1984. The practice of kriging (second edition), *Tech. Rep. No. 263*, Dept. Civ. Eng., Massachusetts Institute of Technology, Cambridge, MA.
- Kueper, B. H., McWhorter, D. B. and E. O. Frind, 1989. The behavior of dense non-aqueous phase liquid contaminants in heterogeneous porous media, In *Proceedings Intl. Sym. on Contaminant Transport in Groundwater*, Stuttgart, Germany.
- Kueper, B. H., E. O. Frind and McWhorter, D. B., 1990. Application of a numerical model and laboratory parameter measurement to the movement of dense, immiscible phase liquids in a heterogeneous sand aquifer, In *Proceedings of Conf. on Subsurface Contamination by Immiscible Fluids*, Calgary, Alberta, Canada.
- Longman, I. M., 1956. Note on a method for computing infinite integrals of oscillatory functions, *Proc. Cambridge Philos. Soc.*, *52(4)*, 764–768.
- MacQuarrie, K. T. B. and E. A. Sudicky, 1989. Simulation of biodegradable organic contaminants in groundwater, 2. Plume behavior in uniform and random flow fields, *Water Resour. Res.*, *26(2)*, 95–110.
- Mantoglou, A. and J. L. Wilson, 1981. Simulation of random fields with the Turning Bands Method, *Tech. Rep. No. 264*, Dept. Civ. Eng., Massachusetts Institute of Technology, Cambridge, MA.
- Mantoglou, A. and J. L. Wilson, 1982. The Turning Bands method for simulation of random fields using line generation by a spectral method, *Water Resour. Res.*, *18(5)*, 1379–1394.
- Matheron, G., 1973. Intrinsic random functions and their applications, *Adv. Appl. Prob.*, *5*, 439–468.

- Mejia, J. and I. Rodriguez-Iturbe, 1974. On the synthesis of random fields from the spectrum: An application to the generation of hydrologic spatial processes, *Water Resour. Res.*, 10(4), 705–711.
- Mizell, S. A., Gutjahr, A. L. and L. W. Gelhar, 1982. Stochastic analysis of spatial variability in two-dimensional steady groundwater flow assuming stationary and non-stationary heads, *Water Resour. Res.*, 18(4), 1053–1067.
- Molissis, D., 1988. Simulation of viscous fingering during miscible displacement in nonuniform porous media, *Ph.D. dissertation, Rice University*, Houston, TX.
- Munoz-Pardo, J. and M. Vaucelin, 1987. Evaluation of different sampling schemes through the simulation of two-dimensional random fields by the Turning Bands method. In *Transactions, American Geophysical Union*, 68(44).
- Neuman, S. P., 1982. Statistical characterization of aquifer heterogeneities: An overview, *Geol. Soc. Am. Spec. Pap.*, 184, 81–102.
- Peterson, D. M. and J. L. Wilson, 1988. Variably saturated flow between streams and aquifers, *New Mexico Water Resour. Res. Inst., Tech. Rep. No. 233*, Las Cruces, NM.
- Rubin, Y. and J. J. Gomez-Hernandez, 1990. A stochastic approach to the problem of upscaling of conductivity in disordered media: Theory and unconditional numerical simulations, *Water Resour. Res.*, 26(4), 691–701.
- Sampier, F. J. and S. P. Neuman, 1989. Estimation of spatial covariance structures by adjoint state maximum likelihood cross-validation, 2 synthetic experiments, *Water Resour. Res.*, 25(3), 363–371.
- Shinozuka, M. and C. M. Jan, 1972. Digital simulation of random processes and its applications, *J. Sound Vib.*, 25(1), 111–128.
- Shumway, R. H., 1988. *Applied Statistical Time Series Analysis*, Prentice Hall, Englewood Cliffs, NJ.
- Smith, L. and R. A. Freeze, 1979. Stochastic analysis of steady state groundwater flow in a bounded domain, 1. One-dimensional simulations, *Water Resour. Res.*, 15(3), 521–528.
- Smith, L. and F. W. Schwartz, 1981. Mass transport 2. Analysis of uncertainty in prediction, *Water Resour. Res.*, 17(2) 351–369.
- Sudicky, E. A., Schellenberg, S. L. and K. T. B. MacQuarrie, 1989. Assessment of the behavior of conservative and biodegradable solutes in heterogeneous porous media, In *Dynamics of Fluids in Hierarchical Porous Formations*, J. H. Cushman (editor), Academic Press, 1990.

Sudicky, E. A. and K. T. B. MacQuarrie, 1989. Behavior of biodegradable organic contaminants in random stationary hydraulic conductivity fields, In *Proceedings Intl. Sym. on Contaminant Transport in Groundwater*, Stuttgart, Germany.

Swain, C. G. and M. S. Swain, 1980. A uniform random number generator that is reproducible, hardware-independent, and fast, *J. Chem. Inf. Comput. Sci.*, 20, 56–58.

Tompson, F. B., Ababou, R. and L. W. Gelhar, 1987. Application and use of the three-dimensional Turning Bands random field generator: Single realization problems, *Tech. Rep. No. 313*, Dept. Civ. Eng., Massachusetts Institute of Technology, Cambridge, MA.

Tompson, F. B., Ababou, R. and L. W. Gelhar, 1989. Implementation of the three-dimensional Turning Bands random field generator, *Water Resour. Res.*, 25(10), 2227–2244.

Vanmarke, E, 1984. *Random Fields, Analysis and Synthesis*, The MIT press, Cambridge, MA.

Wagner, B. J. and S. M. Gorelick, 1989. Reliable aquifer remediation in the presence of spatially variable hydraulic conductivity: From data to design, *Water Resour. Res.*, 25(10), 2211–2225.

Warren, J. E. and H. S. Price, 1961. Flow in heterogeneous porous media, *Soc. Petrol. Eng. J.*, 1, 153–169.

Zimmerman, D. A., Wilson, J. L. and A. L. Gutjahr, 1987. Cosimulation of random fields: Groundwater heads and conductivities, In *Transactions, American Geophysical Union*, 68(44).

Zimmerman, D. A., Gutjahr, A. L. and J. L. Wilson, 1988. A new technique for obtaining stochastic predictions of groundwater flow through heterogeneous porous media: Cosimulation of heads and log-transmissivities using a numerical spectral-perturbation method, *Open File Report 88-4*, Hydrology Program, New Mexico Institute of Mining and Technology, Socorro, NM.

## Appendix A

A Reprint of

*Mantoglou and Wilson, [1982]*



## The Turning Bands Method for Simulation of Random Fields Using Line Generation by a Spectral Method

ARISTOTELIS MANTOGLOU AND JOHN L. WILSON

*Department of Civil Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139*

The turning bands method (TBM) for the simulation of multidimensional random fields is presented. These fields commonly occur in the Monte Carlo simulation of hydrologic processes, particularly groundwater flow and mass transport. The general TBM equations for two- and three-dimensional fields are derived with particular emphasis on the more complicated two-dimensional case. For stationary two-dimensional fields the unidimensional line process is generated by a simple spectral method, a technique which can be generally applied to any two-dimensional covariance function and which is easily extended to anisotropic and areal averaged processes. Theoretically and by example the TBM is shown to be ergodic even for a finite number of lines, and it is demonstrated that it rapidly converges to the true statistics of the field. Guidelines are presented for the selection of model parameters which will be helpful in the design of simulation experiments. The TBM is compared to other methods in terms of cost and accuracy, demonstrating that the TBM is as accurate as and much less expensive than multidimensional spectral techniques and more accurate than the most expensive approaches which use matrix inversion, such as the nearest neighbor approach. The unidimensional spectral technique presented here permits, for the first time, the inexpensive and accurate TBM simulation of any proper two-dimensional covariance function and should be of some help in the stochastic analysis of hydrologic processes.

### INTRODUCTION

Simulation techniques for multidimensional random fields have many important applications in hydrology [see, e.g., Mejia and Rodriguez-Iturbe, 1974; Delhomme, 1979; Smith and Freeze, 1979; Wilson et al., 1979; Freeze, 1980; Smith and Schwartz, 1981], as well as in other geophysical sciences. In hydrology we perform simulations of spatially variable phenomena for one of two reasons. First, because estimations of a field taken from observations are based on a minimization of the estimation error variance (e.g., kriging, as described by Matheron [1973], David [1977], Delhomme [1978], and Journel and Huijbregts [1978]). The resulting estimate is on the average close to the reality, but it is much smoother and does not possess the same fluctuation pattern. We then perform simulations, usually conditional on the observations [see, e.g., David, 1977; Journel and Huijbregts, 1978; Delhomme, 1979], in order to examine typical fluctuation patterns. The second purpose of simulating random fields is to generate input and/or parameter fields for the Monte Carlo simulation of some physical process. These simulated fields are used in a deterministic hydrologic simulation model, often of numerical type, to obtain multiple realizations of the output. From these realizations, we can calculate the statistics of the output and relate them to the statistics of the input and/or parameter fields.

This Monte Carlo approach has recently received attention in the field of groundwater hydrology, where the random parameter field is usually taken as spatially varying hydraulic or storage properties as in the works by Delhomme [1979] or Smith and Freeze [1979], and the output is a field of piezometric heads or even pollutant concentrations [Smith and Schwartz, 1981]. In a parametrically nonlinear system for a sufficiently large number of simulations the Monte Carlo method is superior in accuracy to perturbation or first-order methods, such as those suggested by Dettinger and

Wilson [1981]. For example, inserting an estimated mean (e.g., kriged) hydraulic conductivity parameter field in a deterministic groundwater flow model leads to a biased output. The piezometric head (a first-order expected value) produced from this run is different from the expected value of the piezometric head produced by averaging over an ensemble of Monte Carlo runs (compare Figures 5 and 6 of Delhomme [1979]). Thus Monte Carlo simulation and the use of random field generators have an important place in solving nonlinear stochastic problems that cannot be solved by first-order techniques.

Monte Carlo simulations are also used in rainfall-runoff and other hydrologic models, where the rainfall process [e.g., Mejia and Rodriguez-Iturbe, 1974; Wilson et al., 1979] and/or the physical properties of the watershed [e.g., Freeze, 1980] may be stochastically varying in space.

There are several multidimensional simulation techniques described in the literature. Multivariate or matrix models represent the field only at a number of prespecified discrete points. Assuming stationarity, these models attempt to preserve the covariance of the field between these points [Wilson, 1979]. One of these models, the nearest neighbor approach [Bartlett, 1975], has recently been applied to groundwater flow and transport problems [Smith and Freeze, 1979; Smith and Schwartz, 1981]. The theory of random fields, which characterizes the field not only at the discrete points but also at every other point within the area of interest, was first applied to multidimensional simulation with the development of multidimensional spectral techniques for stationary fields [Shinozuka, 1971; Shinozuka and Jan, 1972; Mejia and Rodriguez-Iturbe, 1974]. Although this approach proves to be less expensive than the matrix approach, it still requires significant computer time to generate a large number of realizations for a Monte Carlo model. Historically, most Monte Carlo simulations of groundwater flow and transport have been performed using the nearest neighbor approach, while the rainfall process is usually simulated using two-dimensional spectral methods.

The turning bands method of multidimensional simulation

Copyright 1982 by the American Geophysical Union.

Paper number 2W0559.  
0043-1397/82/002W-0559\$05.00

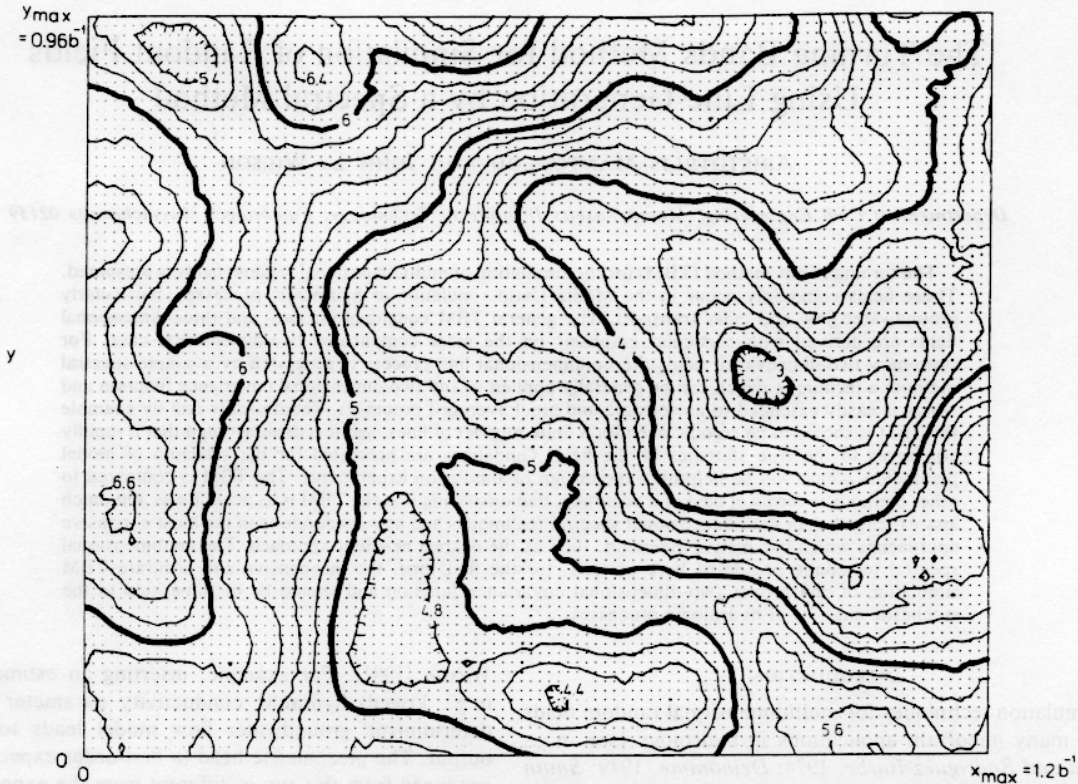


Fig. 1. Typical simulated realization of a two-dimensional stationary process, generated by the turning bands method with spectral method line simulation. The underlying field has a mean of five and an exponential covariance with correlation distance  $b^{-1}$ . NP = 8000,  $L = 16$ ,  $\Delta\zeta = 0.012b^{-1}$ ,  $M = 100$ ,  $\Omega = 40b$ .

is also based on the theory of random fields. Its basic concept is to transform a multidimensional simulation into the sum of a series of equivalent unidimensional simulations. It preserves the statistics of the true field, yet is much less expensive than the other multidimensional methods. A two-dimensional realization of a stationary field generated by the turning bands method is shown in Figure 1. The turning bands method was introduced by Matheron [1973], and has seen extensive application in three-dimensional spatial simulation, particularly with reference to the mining of ore and energy reserves [see, e.g., David, 1977; *Journal and Huijbregts*, 1978]. Two-dimensional spatial simulation, which is often encountered in hydrologic applications, has received much less attention. Chiles [1977] and Delhomme [1979], as members of Matheron's group, briefly examined and employed two-dimensional turning bands simulations in their work but for very special covariance functions not commonly used in hydrology.

As we will show below, two-dimensional spatial processes have more complicated one-dimensional equivalences than do three-dimensional processes and are thus more difficult to simulate by the turning bands method. To overcome these problems we introduce in this paper the spectral equivalences of two-dimensional and one-dimensional processes, permitting for the first time the turning bands simulation of any properly posed two-dimensional covariance function of a stationary random field. We also document the statistical and convergence properties of the turning bands method in general and compare it in terms of cost and accuracy to other multidimensional simulation methods. Before this presenta-

tion a brief review of the concept and properties of random functions is given.

#### REVIEW OF THE PROPERTIES OF STATIONARY RANDOM FUNCTIONS

The concept of a random function is a generalization of the concept of a random variable. If  $\mathbf{x} = (x_1, x_2, \dots, x_n)$  represents a point in  $n$  dimensional space,  $R^n$ , and  $Z(\mathbf{x})$  is a random variable corresponding to point  $\mathbf{x}$ , then we define a random function on  $R^n$  as the set  $\{[\mathbf{x}, Z(\mathbf{x})] | \mathbf{x} \in R^n\}$ . If the dimensionality of the space  $R^n$  is  $n = 2$  or  $n = 3$ , then the random function is usually called a random field. When  $n = 1$ , it is called a line process or unidimensional process. A random function is also called a stochastic process.

The mean function of a stochastic process is defined as

$$m(\mathbf{x}) = E[Z(\mathbf{x})] \quad (1)$$

where  $E[\ ]$  is the expectation operator. If  $E[Z^2(\mathbf{x})]$  is finite for all  $\mathbf{x}$ , we can define the covariance function as

$$\begin{aligned} C(\mathbf{x}_1, \mathbf{x}_2) &= E\{[Z(\mathbf{x}_1) - m(\mathbf{x}_1)][Z(\mathbf{x}_2) - m(\mathbf{x}_2)]\} \\ &= E[Z(\mathbf{x}_1)Z(\mathbf{x}_2)] - m(\mathbf{x}_1)m(\mathbf{x}_2) \end{aligned} \quad (2)$$

where  $\mathbf{x}_1, \mathbf{x}_2 \in R^n$ . A stochastic process is called a 'second-order stationary process' if the following conditions are satisfied:

1. The mean is independent of the position of each point in space  $R^n$ :

$$E[Z(\mathbf{x})] = m(\mathbf{x}) = m \quad \forall \mathbf{x} \in R^n \quad (3)$$



TABLE 1. Some Two-Dimensional  $R^2$  Covariance Functions

Covariance Model	Two Dimensional Covariance Function, $C(r)$	Radial Spectral Density Function, $f(\omega)$
Simple Exponential	$\sigma^2 \exp(-br)$	$\frac{\omega b}{b[1 + (\omega b)^2]^{3/2}}$
Double Exponential	$\sigma^2 \exp(-b^2 r^2)$	$(1/2b)(\omega b) \exp[-(\omega 2b)^2]$
Bessel 1	$\sigma^2 br K_1(br)$	$\frac{2(\omega b)}{b[1 + (\omega b)^2]^2}$
Bessel 2	$\sigma^2 \exp(-b^2 r^2) J_0(ar)$	$\frac{(\omega b)}{2b} \exp\left[-\frac{a^2 + \omega^2}{4b^2}\right] I_0\left[\frac{a\omega}{2b^2}\right]$
Telis	$\sigma^2\{I_0(br) - L_0(br) + br\{I_1(br) - L_{-1}(br)\}\}$	$\frac{4(\omega b)^2}{\pi b[1 + (\omega b)^2]^2}$
Spherical	$\begin{cases} \sigma^2[1 - br(3/2) + (br)^3/2] & 0 \leq br \leq 1 \\ 0 & 1 < br \end{cases}$	$\begin{aligned} & \frac{(\omega b)^2}{4b} J_1\left(\frac{\omega}{b}\right) \left\{ {}_1F_2\left[1; 2, 2; -\frac{(\omega b)^2}{4}\right] \right. \\ & - \frac{2}{3} \times {}_1F_2\left[1; \frac{5}{2}, \frac{5}{2}; -\frac{(\omega b)^2}{4}\right] \\ & \left. + \frac{2}{25} \times {}_1F_2\left[1; \frac{7}{2}, \frac{7}{2}; -\frac{(\omega b)^2}{4}\right] \right\} \end{aligned}$

The Telis function is derived by Mantoglou and Wilson [1982a]. The radial spectral density function for it and for the spherical polynomial function is presented here for the first time, to the best of the authors' knowledge. Radial spectral densities for the others can be found in works by Mejia and Rodriguez-Iturbe [1974] and Matern [1960].  $K_1$  = modified Bessel function of second kind of order one;  $I_0, I_1$  = modified Bessel functions of the first kind of order zero and one, respectively;  $L_0, L_1$  = modified Struve function of order zero and one, respectively;  $J_0, J_1$  = Bessel functions of the first kind of order zero and one, respectively [Abramowitz and Stegun, 1964]. Also,  ${}_1F_2(a; b, c; d)$  is a generalized hypergeometric series as defined by Bailey [1953] or Gradshteyn and Ryzhik [1980].

2. The covariance function depends only on the vector difference  $(x_1 - x_2)$  and not on each particular vector  $x_1, x_2$ :

$$C(x_1, x_2) = C(x_1 - x_2) = C(h) \tag{4}$$

where  $h = x_1 - x_2$ . The terms second-order stationary, broad stationary, and wide sense stationary are interchangeable. In the case where  $x$  is a space parameter, the term broad sense homogeneity is often used instead. All the processes dealt with in this paper are assumed to be second order stationary.

A second-order stationary process is called isotropic if the covariance function does not depend on the direction  $h = x_1 - x_2$  of the distance vector, but only on the vector length  $|h|$ . Then we can write

$$C(h) = C(r) \tag{5}$$

where  $r = |h|$ . Some proper two-dimensional isotropic covariance functions are presented in Table 1, where  $r$  is distance between any two points in  $R^n$  and  $\sigma^2$  is the variance of the process. (See Matern [1960] or Veneziano [1978] for properties of proper covariance functions.) The exponential, double exponential, and spherical functions can be used in one, two, or three dimensions ( $n = 1, 2, \text{ or } 3$ ) [see Matern, 1960; Delhomme, 1978; Veneziano, 1978]. The Bessel functions and Telis function are used only in two dimensions,  $R^2$  [see Mejia and Rodriguez-Iturbe, 1974; Mantoglou and Wilson, 1982a, respectively]. The parameter  $b^{-1}$  in each function has units of length and is sometimes called the correlation length. The radial spectral density function  $f(\omega)$  of a two-dimensional isotropic process is defined later.

THE TURNING BANDS METHOD

It is assumed that the field to be simulated is second-order stationary and isotropic. It is also assumed that at each point the values of the field are normally distributed and have zero mean. If this is not the case, we can usually make a transformation to Gaussian and then subtract the mean. It is also assumed that the covariance  $C(r)$  of the field we want to simulate is known.

Instead of simulating the two- or three-dimensional field directly, in the turning bands method we perform simulations along several lines, using a unidimensional covariance function that corresponds to the given two- or three-dimensional one. Then at each point of the two- or three-dimensional field a weighted sum of the corresponding values of the line processes is assigned.

Let  $P$  represent the two- or three-dimensional field we want to simulate by generating values at discrete points in it. A two-dimensional example is shown in Figure 2. Choose an arbitrary origin 0 in  $R^n$  and generate lines such that the corresponding direction vectors  $u$  are uniformly distributed on the unit circle or sphere in two or three dimensions, respectively. In the two-dimensional case, for example, the angle  $\theta_i$  formed between a line  $i$  and a fixed  $x$  axis (Figure 2) is uniformly distributed between 0 and  $2\pi$ . Along each line  $i$ , generate a second-order stationary unidimensional discrete process having zero mean and covariance function  $C_i(\zeta)$ , where  $\zeta$  is the coordinate on line  $i$ . We will derive later the correspondence between  $C_i(\zeta)$  and the covariance function of the field. Onto line  $i$ , orthogonally project those points of the field where we want to generate values, and assign to

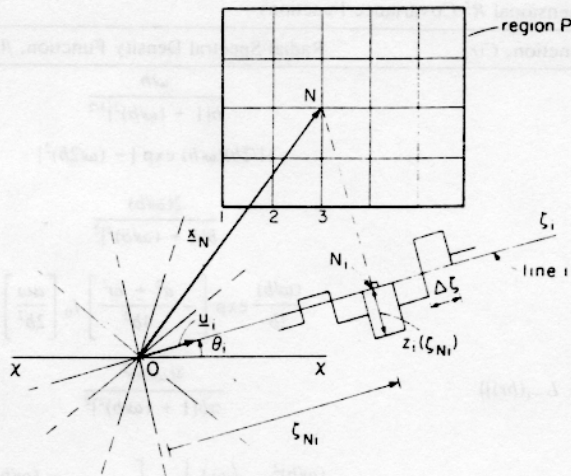


Fig. 2. Schematic representation of the field  $P$  and the turning bands lines  $i$ .

them the corresponding values of the one dimensional discrete process. If  $N$  is a point of the region having location vector  $\mathbf{x}_N$ , then the assigned value from line  $i$  will be  $z_i(\zeta_{Ni})$  where  $\zeta_{Ni} = \mathbf{x}_N \cdot \mathbf{u}_i$  is the projection of the vector  $\mathbf{x}_N$  onto line  $i$  (see Figure 2),  $\mathbf{u}_i$  the unit vector on line  $i$ , and  $\mathbf{x}_N \cdot \mathbf{u}_i$  represents the inner product of the vectors  $\mathbf{x}_N$  and  $\mathbf{u}_i$ . Take  $L$  lines such as  $i$ . For each line generate an independent unidimensional realization using  $C_1(\zeta)$  as the covariance function. Then at every point  $N$  of the region, there are  $L$  assigned values  $z_i(\zeta_{Ni}) = z_i(\mathbf{x}_N \cdot \mathbf{u}_i)$ , where  $i = 1, \dots, L$ , from the unidimensional realizations. Finally, assign to the point  $N$  the value  $z_s(\mathbf{x}_N)$  given by

$$z_s(\mathbf{x}_N) = \frac{1}{\sqrt{L}} \sum_{i=1}^L z_i(\mathbf{x}_N \cdot \mathbf{u}_i) \quad (6)$$

as the realization of the two- or three-dimensional random field. The subscript  $s$  represents the term 'simulated' or 'synthetic.'

The line process is generated discretely. If we draw lines or planes perpendicular to the line at the ends of each discretized segment, a set of bands is defined (Figure 2). As the lines turn, the bands defined above also turn. Thus the method was given the name 'turning bands method' (TBM) by Matheron [1973].

The field given by (6) has zero mean. The question that arises is 'what is the form of the unidimensional covariance function  $C_1(\zeta)$  so that the field defined by (6) has the imposed two- or three-dimensional covariance function  $C(r)$ ?' Take two points of the field having location vectors  $\mathbf{x}_1, \mathbf{x}_2$ , respectively. The simulated values corresponding to these points are given by (6) and the covariance function of the simulated field is

$$C_s(\mathbf{x}_1, \mathbf{x}_2) = E[Z_s(\mathbf{x}_1) Z_s(\mathbf{x}_2)] \\ = \frac{1}{L} \sum_{i=1}^L \sum_{j=1}^L E[Z_i(\mathbf{x}_1 \cdot \mathbf{u}_i) Z_j(\mathbf{x}_2 \cdot \mathbf{u}_j)] \quad (7)$$

Because the unidimensional realizations along two different lines are independent, the expected value  $E[Z_i(\mathbf{x}_1 \cdot \mathbf{u}_i) Z_j(\mathbf{x}_2 \cdot \mathbf{u}_j)]$  will be zero unless  $i = j$ . Thus this expression reduces to

$$C_s(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{L} \sum_{i=1}^L E[Z_i(\mathbf{x}_1 \cdot \mathbf{u}_i) Z_i(\mathbf{x}_2 \cdot \mathbf{u}_i)] \\ = \frac{1}{L} \sum_{i=1}^L C_1(\mathbf{h} \cdot \mathbf{u}_i) \quad (8)$$

where  $\mathbf{h} = \mathbf{x}_2 - \mathbf{x}_1$ . The expected value  $E[Z_i(\mathbf{x}_1 \cdot \mathbf{u}_i) Z_i(\mathbf{x}_2 \cdot \mathbf{u}_i)]$  represents the covariance of the one-dimensional process on line  $i$  between points  $\mathbf{x}_1 \cdot \mathbf{u}_i$  and  $\mathbf{x}_2 \cdot \mathbf{u}_i$ , which is written as  $C_1(\mathbf{h} \cdot \mathbf{u}_i) = E[Z_i(\mathbf{x}_1 \cdot \mathbf{u}_i) Z_i(\mathbf{x}_2 \cdot \mathbf{u}_i)]$ , assuming that the unidimensional process is second-order stationary. Because the vector  $\mathbf{u}_i$  is uniformly distributed over the unit circle or sphere, the right-hand side of (8) is only a function of  $|\mathbf{h}|$  for large  $L$ . This means that the obtained process is wide sense stationary and isotropic, so that we can write

$$C_s(\mathbf{x}_1, \mathbf{x}_2) = C_s(\mathbf{h}) = C_s(r) = \frac{1}{L} \sum_{i=1}^L C_1(\mathbf{h} \cdot \mathbf{u}_i) \quad (9)$$

where  $r = |\mathbf{h}|$ . For  $L \rightarrow \infty$ , by using the law of large numbers this becomes

$$C_s(r) = \lim_{L \rightarrow \infty} \left\{ \frac{1}{L} \sum_{i=1}^L C_1(\mathbf{h} \cdot \mathbf{u}_i) \right\} = E[C_1(\mathbf{h} \cdot \mathbf{u})] \\ = \int_c C_1(\mathbf{h} \cdot \mathbf{u}) f(\mathbf{u}) d\mathbf{u} \quad (10)$$

where  $c$  represents the unit circle or sphere,  $f(\mathbf{u})$  is the probability density function of  $\mathbf{u}$  which becomes  $1/2\pi$  or  $1/4\pi$  in two- or three-dimensional cases, respectively, and  $d\mathbf{u}$  is the differential length or area at the end of vector  $\mathbf{u}$ . Equation (10) then gives, for the two-dimensional case ( $n = 2$ ),

$$C_s(r) = \frac{1}{2\pi} \int_{\text{unit circle}} C_1(\mathbf{h} \cdot \mathbf{u}) d\mathbf{u} \quad (11)$$

while for the three-dimensional field ( $n = 3$ ) we have

$$C_s(r) = \frac{1}{4\pi} \int_{\text{unit sphere}} C_1(\mathbf{h} \cdot \mathbf{u}) d\mathbf{u} \quad (12)$$

In the following we examine the cases of three- and two-dimensional fields separately.

### Three-Dimensional Fields

For examples, see *Journal and Huigbregts* [1978]. Because of the second-order stationarity and isotropy of the process, without loss of generality we can define orthogonal  $(x, y, z)$  axes with origin at the point  $\mathbf{x}_1$  and with the  $z$  axis in the direction of the vector  $\mathbf{h} = \mathbf{x}_2 - \mathbf{x}_1$ , as shown in Figure 3. The unit sphere where the vector  $\mathbf{u}$  ends is also shown. In spherical coordinates  $\mathbf{h} \cdot \mathbf{u} = r \cos \phi$ , where  $r = |\mathbf{h}|$ , and  $d\mathbf{u} = \sin \phi d\phi d\theta$ . The integral (12) is then written as

$$C_s(r) = \frac{1}{4\pi} \int_0^{2\pi} \int_0^\pi C_1(r \cos \phi) \sin \phi d\phi d\theta \\ = \frac{1}{r} \int_0^r C_1(\zeta) d\zeta \quad (13)$$

taking advantage of the symmetry of  $C_1$  and introducing the line coordinate  $\zeta = r \cos \phi$  oriented in the direction of  $u$ . Differentiating and changing notation leads to the relationship between one- and three-dimensional covariances:

$$C_1(\zeta) = \frac{d}{d\zeta} [\zeta C(\zeta)] \tag{14}$$

where we have set  $C_1(\zeta) = C(\zeta)$  in order to preserve the known three-dimensional covariance during simulation. This simple relationship can be easily applied for the calculation of  $C_1(\zeta)$ . For an exponential three-dimensional covariance  $C(r) = \sigma^2 \exp(-br)$ , the corresponding one-dimensional covariance is  $C_1(\zeta) = \sigma^2(1 - b\zeta) \exp(-b\zeta)$ , commonly called the 'hole function.' For a double exponential three-dimensional covariance  $C(r) = \sigma^2 \exp(-b^2r^2)$ , the corresponding one-dimensional covariance is  $C_1(\zeta) = \sigma^2(1 - 2b^2\zeta^2) \exp(-b^2\zeta^2)$ . For a spherical three-dimensional covariance  $C(r) = \sigma^2[1 - 3br/2 + (br)^3/2]$  for  $0 \leq br \leq 1$  and  $C(r) = 0$  for  $br > 1$ , the corresponding one-dimensional covariance is  $C_1(\zeta) = \sigma^2[1 - 3b\zeta + 2(b\zeta)^3]$  for  $0 \leq b\zeta \leq 1$  and  $C_1(\zeta) = 0$  for  $b\zeta > 1$ . All of these unidimensional covariance functions share the property that they can be modeled as a moving average (MA) process, leading to a simple and direct way of generating realizations along the lines [Journel and Huijbregts, 1978; Mantoglou and Wilson, 1981, 1982a]. It is not possible to find similarly simple expressions for  $C_1(\zeta)$  in the case of two-dimensional fields.

*Two-Dimensional Fields*

Define orthogonal axes  $(x, y)$  in the plane of the field, with origin at point  $x_1$  and the  $y$  axis in the direction of the vector  $h = x_2 - x_1$  as shown in Figure 4. In polar coordinates we can write  $h \cdot u = r \sin \theta$  and  $du = d\theta$ . Equation (11) then becomes

$$C_r(r) = \frac{1}{2\pi} \int_0^{2\pi} C_1(r \sin \theta) d\theta = \frac{2}{\pi} \int_0^r \frac{C_1(\zeta)}{(r^2 - \zeta^2)^{1/2}} d\zeta \tag{15}$$

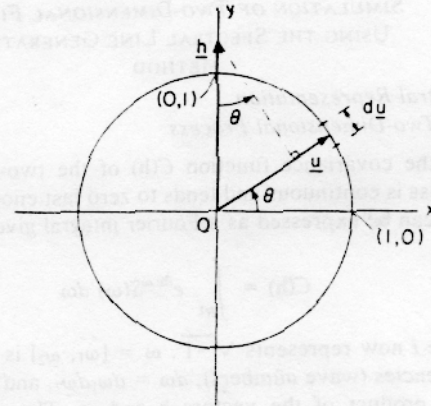


Fig. 4. Definition sketch for the two-dimensional case, showing unit circle.

where  $\zeta = r \sin \theta$  and we have noted that  $C_1$  is an even function. Substituting  $C_1(r) = C(r)$ , in order to preserve the known covariance, we get

$$\int_0^r \frac{C_1(\zeta)}{(r^2 - \zeta^2)^{1/2}} d\zeta = \frac{\pi}{2} C(r) \tag{16}$$

This equation relates the two dimensional covariance function  $C(r)$  to the corresponding unidimensional  $C_1(\zeta)$  along the turning band lines. It is an integral equation in which  $C_1(\zeta)$  cannot be directly expressed as a function of  $C(r)$ . Particular solutions though can be found for certain two-dimensional covariance functions, but these lead to unidimensional covariances that cannot be modeled easily as an MA or autoregressive (AR) process [Mantoglou and Wilson, 1981, 1982a]. To circumvent this difficulty, an expression for the spectral density function of the one-dimensional process as a function of the radial spectral density function of the two-dimensional process is derived below. The line process can then be generated easily using a spectral method (such as those of Rice [1954] or Shinozuka and Jan [1972]). If  $C(r)$  is a proper two- or three-dimensional covariance function, it can be shown by means of Bochner's theorem that the function  $C_1(\zeta)$  given by (14) or (16) is a positive definite function in one dimension and can thus be used as a unidimensional covariance function.

*Line Distribution*

Equations (10), (11), and (12) are obtained in the limit as the number of lines goes to infinity. The lines are assumed to be randomly oriented, as taken from a uniform distribution on the unit circle or sphere. It can be easily shown that these TBM equations are also obtained by spacing the lines evenly on the unit circle or sphere, with prescribed directions. If the same line orientations are maintained during a sequence of simulations, both methods are ergodic, but the simulated covariance converges much faster to the theoretical function for the even spacing approach, as we prove later, so this approach is preferred. In three dimensions, experience has shown [Journel and Huijbregts, 1978] that a group of 15 lines, joining the midpoints of the opposite edges of a regular icosahedron, is adequate for typical applications. For two-dimensional fields encountered in hydrologic applications, the focus of this paper, our experience is that 4-16 lines should be sufficient, depending on the accuracy desired (see section on accuracy later).

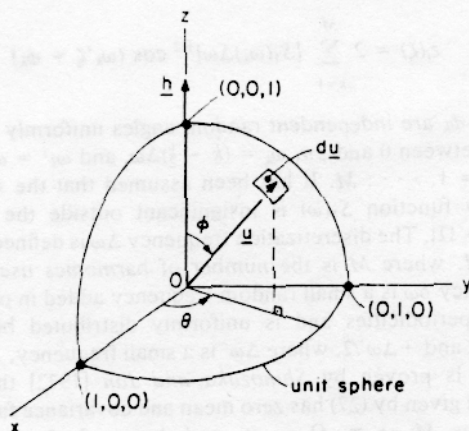


Fig. 3. Definition sketch for the three-dimensional case, showing the unit sphere.



SIMULATION OF TWO-DIMENSIONAL FIELDS  
USING THE SPECTRAL LINE GENERATION  
METHOD

Spectral Representation  
of a Two-Dimensional Process

If the covariance function  $C(\mathbf{h})$  of the two-dimensional process is continuous and tends to zero fast enough as  $|\mathbf{h}| \rightarrow \infty$ , it can be expressed as a Fourier integral given by

$$C(\mathbf{h}) = \int_{R^2} e^{i\mathbf{h}\cdot\boldsymbol{\omega}} S(\boldsymbol{\omega}) d\boldsymbol{\omega} \quad (17)$$

where  $i$  now represents  $\sqrt{-1}$ ,  $\boldsymbol{\omega} = [\omega_1, \omega_2]$  is a vector of frequencies (wave numbers),  $d\boldsymbol{\omega} = d\omega_1 d\omega_2$ , and  $\mathbf{h} \cdot \boldsymbol{\omega}$  is the inner product of the vectors  $\mathbf{h}$  and  $\boldsymbol{\omega}$ . The nonnegative function  $S(\boldsymbol{\omega}) = S(\omega_1, \omega_2)$  is the spectral density function of the two-dimensional process and is given by the Fourier transform of  $C(\mathbf{h})$ :

$$S(\boldsymbol{\omega}) = \frac{1}{(2\pi)^2} \int_{R^2} e^{-i\mathbf{h}\cdot\boldsymbol{\omega}} C(\mathbf{h}) d\mathbf{h} \quad (18)$$

If the field is isotropic, then  $S(\boldsymbol{\omega}) = S(\omega)$ , in which  $\omega = |\boldsymbol{\omega}|$ . The Fourier transform pair (17) and (18) then becomes [Shoenberg, 1938; Matern, 1960]

$$C(r) = \sigma^2 \int_0^\infty f(\omega) J_0(\omega r) d\omega \quad (19)$$

$$f(\omega) = \frac{\omega}{\sigma^2} \int_0^\infty C(r) J_0(\omega r) r dr \quad (20)$$

where  $r = |\mathbf{h}|$ ,  $J_0(\cdot)$  is a Bessel function of first kind of order zero, and  $f(\omega)$  is the radial spectral density function of the two dimensional isotropic process defined by

$$f(\omega) = \frac{1}{\sigma^2} \int_{c_\omega} S(\boldsymbol{\omega}) d\boldsymbol{\omega} = \frac{1}{\sigma^2} S(\omega) \int_{c_\omega} d\boldsymbol{\omega} = \frac{2\pi\omega S(\omega)}{\sigma^2} \quad (21)$$

where  $c_\omega$  is a circle of radius  $\omega$  and  $d\boldsymbol{\omega}$  is the differential length on circle  $c_\omega$ . If the two-dimensional isotropic covariance function  $C(r)$  is known, we can use (20) to calculate the corresponding radial spectral density function. Some examples for common two-dimensional covariance functions are given in Table 1. Then these  $f(\omega)$  are related to the spectral density function  $S(\omega)$  of the two dimensional process through (21).

Spectral Representation  
of the Unidimensional Process

Let  $S_1(\omega)$  be the spectral density function of the unidimensional process having covariance function  $C_1(\zeta)$ . Then in one dimension the Fourier representation equivalent to (17) is

$$C_1(\zeta) = \int_{-\infty}^{\infty} e^{i\omega\zeta} S_1(\omega) d\omega = 2 \int_0^\infty \cos(\omega\zeta) S_1(\omega) d\omega \quad (22)$$

where the frequency (wave number)  $\omega$  is a scalar, and  $S_1(\omega)$  is real, symmetric, and positive. For the two-dimensional geometry of Figure 4,  $\zeta = r \cos \theta$ , since  $\zeta$  is oriented in the same direction as  $\mathbf{u}$ . Substituting this Fourier representation

of  $C_1(\zeta)$  into the TBM (15) setting  $C_1(r) = C(r)$  and changing the order of integration leads to

$$C(r) = \frac{4}{\pi} \int_0^\infty S_1(\omega) \left\{ \int_0^{\pi/2} \cos(\omega r \sin \theta) d\theta \right\} d\omega \quad (23)$$

From *Gradshteyn-Ryzhik* [1980, section 8.41] the integral inside the brackets  $\{ \}$  becomes  $(\pi/2)J_0(\omega r)$ . Thus (23) can be written as

$$C(r) = 2 \int_0^\infty S_1(\omega) J_0(\omega r) d\omega \quad (24)$$

Applying a Hankel transform to this equation gives

$$S_1(\omega) = \frac{1}{2} \omega \int_0^\infty C(r) J_0(\omega r) r dr \quad (25)$$

Comparing (25) and (20), we obtain

$$S_1(\omega) = \frac{\sigma^2}{2} f(\omega) \quad (26)$$

This means that the spectral density function of the unidimensional process along the turning bands lines is simply given by one half of the radial spectral density function of the two-dimensional process multiplied by the variance. We can use (26) to derive the spectral density function of the unidimensional process for various two-dimensional covariance functions, using previously derived radial spectral density functions (as in Table 1). With this approach we can now simulate any proper two-dimensional covariance function!

Generation of the Line Process

After obtaining the spectral density function of the unidimensional process, we can easily generate the process along the turning bands lines using any spectral method. Here we use the classical method proposed by *Rice* [1954] and modified by *Shinozuka and Jan* [1972]. For unidimensional processes it is proven in the second reference that this method is superior to the corresponding method of *Shinozuka* [1971] or *Mejia and Rodriguez-Iturbe* [1974] in terms of accuracy and cost. If the unidimensional covariance function is  $C_1(\zeta)$  and the corresponding spectral density function, obtained in (26), is  $S_1(\omega)$ , then the unidimensional process on line  $i$  can be generated by

$$z_i(\zeta) = 2 \sum_{k=1}^M [S_1(\omega_k) \Delta\omega]^{1/2} \cos(\omega_k' \zeta + \phi_k) \quad (27)$$

where  $\phi_k$  are independent random angles uniformly distributed between 0 and  $2\pi$ ,  $\omega_k = (k - \frac{1}{2})\Delta\omega$ , and  $\omega_k' = \omega_k + \delta\omega$  for  $k = 1, \dots, M$ . It has been assumed that the spectral density function  $S_1(\omega)$  is insignificant outside the region  $[-\Omega, +\Omega]$ . The discretization frequency  $\Delta\omega$  is defined as  $\Delta\omega = \Omega/M$ , where  $M$  is the number of harmonics used. The frequency  $\delta\omega$  is a small random frequency added in order to avoid periodicities and is uniformly distributed between  $-\Delta\omega'/2$  and  $+\Delta\omega'/2$ , where  $\Delta\omega'$  is a small frequency,  $\Delta\omega' \ll \Delta\omega$ . It is proven by *Shinozuka and Jan* [1972] that the process given by (27) has zero mean and covariance function  $C_1(\zeta)$ , as  $M \rightarrow \infty$ ,  $\Omega \rightarrow \infty$ , and  $\Delta\omega \rightarrow 0$ . In practical applications we have  $M < \infty$ ,  $\Omega < \infty$ , and  $\Delta\omega > 0$ , so that some error is introduced. The magnitude of this error will be

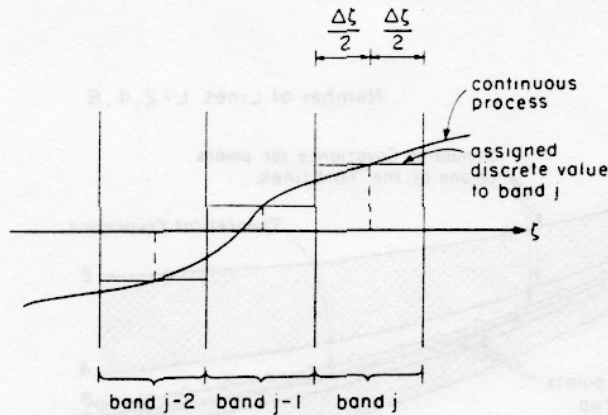


Fig. 5. Continuous and discrete processes on a turning bands line.

discussed later. It is also noted that this process is strictly ergodic, even for a finite number of lines, and Gaussian. In the turning bands method (27) is used to generate values at discrete points. These points are chosen to be the middle points of the segments defined by the bands along each line. The same value is assigned to the entire segment or band as shown in Figure 5. By using the spectral method, we could also generate directly values on the turning bands lines at the projections of the points of the field. This approach would eliminate the error introduced by the generation of the line process discretely in 'bands;' however, the cost increases rapidly, since the number of points generated along the lines is equal to the number of points of the field. Although we tested this approach, it was abandoned because of its high cost in favor of preassigned discrete point (band) generation. Finally, we note that more efficient unidimensional spectral methods, such as the fast Fourier transform, can be used instead of the method of (27).

ACCURACY OF THE TURNING BANDS METHOD FOR SIMULATION OF TWO-DIMENSIONAL FIELDS

Sources of Error

There are two major types of error introduced by the use of simulation. The first is due to estimation of the statistics of the underlying random field from a limited data set. We call this a 'model-to-reality fitting error;' it includes violated assumptions on isotropy, stationarity, constant mean, etc., as well as errors in the estimation of the covariance function. We ignore this error here, although it is undoubtedly important. The second type of error is called 'within simulation error' and includes simulation model error and simulation estimation error. Simulation model errors are caused by discretizations or approximations used in the TBM, such as the finite number of lines ( $L$ ), the discretization along the lines (band width  $\Delta\xi$ ), the discretization of the spectrum of the line process ( $\Delta\omega$ ), and the finite number of harmonics used in the generation of the unidimensional process ( $M$ ). These errors, which are described immediately below, can be reduced by proper design of the simulation runs. Simulation estimation error is introduced by the use of a finite number of simulations. As the number of simulations increases this estimation error tends to zero, as shown later. Our examination of the simulation model error will focus first on the number of lines for two different line spacings:

random (taken from a uniform distribution) and evenly spaced at prespecified locations.

Accuracy as a Function of the Number of Lines

*Lines randomly distributed.* This process preserves the mean, as can easily be seen from (6). If the uniformly distributed random lines are generated only once at the beginning of the simulations, and if the same lines are used for all the simulation runs, the simulated covariance function is given by (9). This covariance is valid as either an ensemble or spatial average as long as the unidimensional process is ergodic (as is the process generated by Rice's [1954] method). Consequently, the generated two-dimensional process is ergodic. The error in the covariance function due to a finite number of lines is  $\epsilon = C_s(r) - C(r)$ , where  $\epsilon$  tends to zero for large  $L$ . The expected value of the error is zero,  $E[\epsilon] = 0$ , since

$$E[C_s(r)] = E\left[\frac{1}{L} \sum_{i=1}^L C_1(\mathbf{h} \cdot \mathbf{u}_i)\right] = E[C_1(\mathbf{h} \cdot \mathbf{u})] = C(r) \quad (28)$$

where  $C_1(\cdot)$  satisfies (16) or (11). The variance of the error is given by

$$\begin{aligned} \text{Var}[\epsilon] &= \text{Var}[C_s(r)] = E\left[\left\{\frac{1}{L} \sum_{i=1}^L C_1(\mathbf{h} \cdot \mathbf{u}_i) - C(r)\right\}^2\right] \\ &= \frac{1}{L^2} E\left[\sum_{i=1}^L \sum_{j=1}^L \{C_1(\mathbf{h} \cdot \mathbf{u}_i) - C(r)\} \cdot \{C_1(\mathbf{h} \cdot \mathbf{u}_j) - C(r)\}\right] \\ &= \frac{1}{L} \{E[C_1^2(\mathbf{h} \cdot \mathbf{u})] - C^2(r)\} \quad (29) \end{aligned}$$

because  $\mathbf{u}_i$  and  $\mathbf{u}_j$  are uncorrelated. Consequently, the standard deviation of the covariance error is a function of  $r$  and tends to zero as  $1/\sqrt{L}$ :

$$\sigma_\epsilon(r) = \frac{1}{\sqrt{L}} \{E[C_1^2(\mathbf{h} \cdot \mathbf{u})] - C^2(r)\}^{1/2} \quad (30)$$

The rate of convergence is very slow for this approach of randomly generating lines, and thus it is never used. We could also generate different sets of randomly distributed lines for each run. Thus we obtain the true covariance as an ensemble average but the covariance given by (9) for a spatial average. This alternative approach yields a nonergodic process and was not pursued.

*Lines evenly spaced.* The theoretical two-dimensional covariance function is given by (15) with  $C_s(r) = C(r)$ , and the TBM simulated covariance is given by (9) in which  $\mathbf{h} \cdot \mathbf{u}_i = r \sin \theta_i$ . The covariance error for evenly spaced lines on the unit circle becomes

$$\begin{aligned} \epsilon &= C_s(r) - C(r) \\ &= \frac{1}{L} \sum_{i=1}^L C_1(r \sin \theta_i) - \frac{1}{\pi} \int_0^\pi C_1(r \sin \theta) d\theta \quad (31) \end{aligned}$$

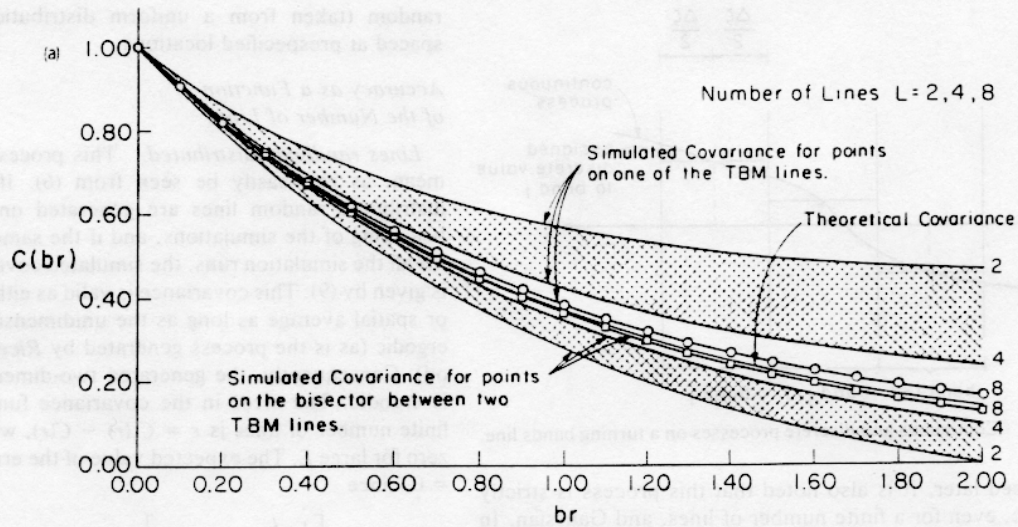


Fig. 6. Covariance error envelop due to a finite number of lines for evenly spaced lines and  $\sigma^2 = 1$ . The theoretical two-dimensional covariance is exponential. (a)  $L = 2, 4, 8$ . (b)  $L = 16$ .

A first-order approximation of the limits of this error is derived in the appendix, in which it is assumed that  $L$  is an even number:

$$\frac{-\pi\sigma^2 Kbr}{12L^2} \leq \epsilon \leq \frac{\pi\sigma^2 Kbr}{6L^2} \quad (32)$$

where  $K = (-1/\sigma^2 b)[dC_1(\zeta)/d\zeta]_{\zeta=0}$  is a positive constant that depends on the unidimensional covariance function. If, for example,  $C_1(\zeta)$  is a hole function,  $C_1(\zeta) = \sigma^2(1 - b\zeta) \exp(-b\zeta)$ , then  $K = 2$ . For a two-dimensional exponential covariance,  $C(r) = \sigma^2 \exp(-br)$ , the corresponding unidimensional covariance (36), yields  $K = 1.6$ . The bounds of the error are obtained for points lying a distance  $r$  apart. The lower bound is for points along the bisector of two turning bands lines, while the upper bound is for points lying on one of the lines. The error estimates show that for evenly spaced lines the simulated covariance asymptotically converges to the true covariance rapidly as  $1/L^2$ . They also indicate that the error increases linearly with distance  $r$ , at least for small  $r$ . For large  $r$  the higher order terms neglected in (32) (see appendix) are important, and the error has a finite limit as  $r \rightarrow \infty$ . For a fixed number of lines  $L$  this error can be easily shown to be, for points on a TBM line:

$$\lim_{r \rightarrow \infty} \epsilon = \sigma^2/L \quad (33a)$$

and for points on bisector between two lines:

$$\lim_{r \rightarrow \infty} \epsilon = 0 \quad (33b)$$

so that the error is always between  $0 \leq \epsilon \leq \sigma^2/L$  as  $r \rightarrow \infty$ . When the angle of the line connecting the simulated points is uniformly distributed, the expected covariance error lies halfway between the bounds of (32). Then the simulation is biased since the expected error is  $E[\epsilon] = \pi\sigma^2 Kbr/24L^2$ . The standard deviation of the error about this biased mean is

$$\sigma_{\epsilon'} = \{\text{Var}[\epsilon]\}^{1/2} = \frac{3\pi\sigma^2 Kbr}{12^{3/2}L^2} = \frac{0.227\sigma^2 Kbr}{L^2} \quad (34)$$

while the square root of the error second moment about the theoretical covariance is

$$\sigma_{\epsilon} = \{E[(C_s(r) - C(r))^2]\}^{1/2} = \frac{7\pi\sigma^2 Kbr}{36L^2} = \frac{0.231\sigma^2 Kbr}{L^2} \quad (35)$$

The bias and the standard deviation of the simulation error rapidly tend to zero as  $1/L^2$ . Even though they are slightly biased, evenly spaced lines are preferred in practice because of their rapid convergence and are used in all of our simulations.

When the turning bands lines are evenly spaced on the unit circle, the angle  $\theta_1$  of the first line is specified ( $\theta_1 = 0$ ). An alternative is to uniformly distribute the angle  $\theta_1$  of the first line between 0 and  $\pi/L$ . It can be shown that the resulting process better preserves the ensemble covariance but that the error in the spatial estimation of the covariance remains unchanged. This means that this process is no longer ergodic, at least for smaller  $L$ . Prespecifying  $\theta_1 = 0$ , as done for the following examples in this paper, assures ergodicity for all values of  $L$ .

*How many lines?* In works by Mantoglou and Wilson [1981, 1982a] the integral equation (16) is solved to yield  $C_1(\zeta)$  for some particular two-dimensional covariance functions often encountered in hydrology. Using these expressions for  $C_1(\zeta)$  and (9) with  $\mathbf{h} \cdot \mathbf{u}_i = r \sin \theta_i$  for the simulated covariance  $C_s(r)$  as a function of the number of lines, we can directly compare our approximate  $C_s(r)$  to the theoretical covariance function  $C(r)$ . For the exponential function,  $C(r) = \sigma^2 \exp(-br)$ , the corresponding one-dimensional covariance is [Mantoglou and Wilson, 1981, 1982a]

$$C_1(\zeta) = \sigma^2 \left\{ 1 - \frac{\pi}{2} b\zeta [I_0(b\zeta) - L_0(b\zeta)] \right\} \quad (36)$$

where  $I_0$  is a Bessel function of the first kind of order zero and  $L_0$  is a modified Struve function of order zero. Figure 6 shows the theoretical covariance function and the envelope within which the simulated covariance lies for different number of evenly spaced lines  $L$ . The theoretical covariance function has been normalized by dividing by the variance.



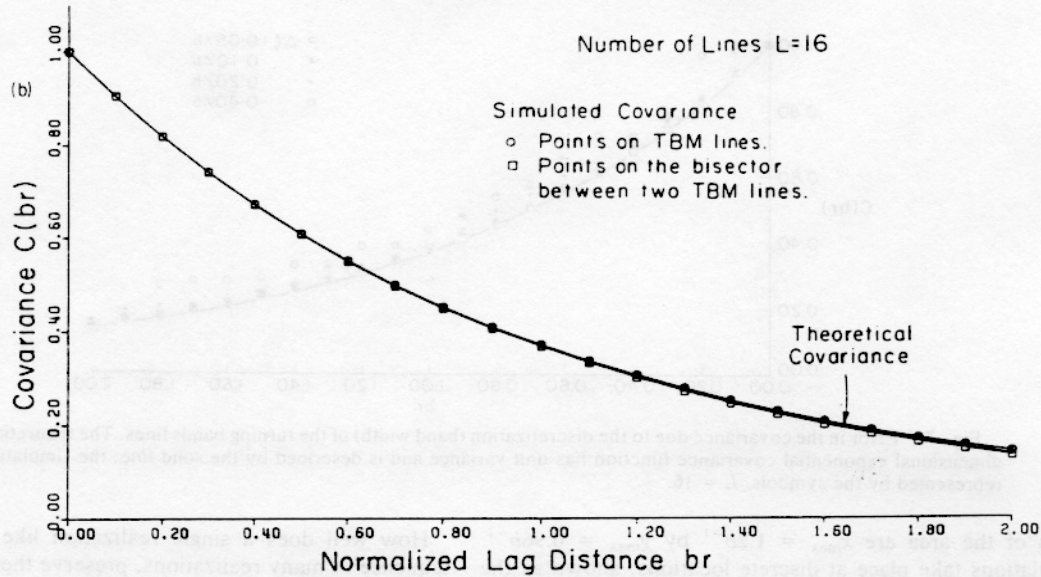


Fig. 6. (continued)

and the lag distance  $r$  has been normalized by dividing by the correlation length  $b^{-1}$ . We will refer to the resulting normalized process as a unit variance process in all of the examples that follow. The upper envelope represents points that lie on one of the turning bands lines, and the lower envelope represents points that lie on the bisector between two lines, giving a result similar to (32). As seen from these curves, the simulated covariance rapidly approaches the theoretical covariance as  $L$  increases.  $L = 16$  lines provides a very accurate representation of the process, while  $L = 4$  is adequate. These curves and/or the error estimate (32) can be used to guide the selection of  $L$ . For the exponential covariance the maximum error in (32) is  $|e| \leq 0.84\sigma^2 br/L^2$ , where  $r$  represents the largest distance between simulated points in the field. For  $r_{\max} = 2/b$  (two correlation lengths) and  $L = 8$ , this error becomes  $|e| \leq 0.026\sigma^2$  or 2.6% of the point variance (see also  $L = 8$  case in Figure 6a).

*Other Sources of Model Error*

*Discretization along the lines.* The effect of discretization length (band width)  $\Delta\zeta$  on the model accuracy can also be tested via (9) and (36), where now the argument in the function  $C_1(\cdot)$  is  $\zeta_i = \|r \sin \theta / \Delta\zeta\| \cdot \Delta\zeta$ , in which  $\|\xi\|$  represents the greatest integer such that  $\|\xi\| \leq \xi$ . For  $L = 16$ , the error due to finite number of lines is very small, as we have already seen in Figure 6, so that we can assume that any additional error in the experiment is due to discretization. Using a unit variance exponential covariance function, looking at points lying on one of the turning bands lines, and taking the origin of the turning bands lines at the midpoint between two simulated points, yields the results shown in Figure 7. This figure compares the theoretical covariance to simulated covariances obtained with band widths  $\Delta\zeta = 0.05b^{-1}$ – $0.40b^{-1}$ , where  $b^{-1}$  represents the correlation length. As shown in the figure, the distance between simulated points is  $\Delta x = 0.01b^{-1}$ ; thus  $\Delta\zeta/\Delta x$  varies from 0.5 to 4.0. For  $\Delta\zeta = 0.05b^{-1}$  or  $0.1b^{-1}$ , the accuracy is excellent; for  $\Delta\zeta = 0.2b^{-1}$ , the accuracy is adequate; and for  $\Delta\zeta = 0.4b^{-1}$ , the accuracy is relatively poor. In practical applications, when we generate a grid of, say, rectangular grid spacing  $\Delta x, \Delta y$ ,

we take  $\Delta\zeta < \min(\Delta x, \Delta y)$  in order to avoid problems, such as where to choose the origin of the turning bands lines.

*Approximations in the spectral method of line generation.* The accuracy of the TBM finally depends on the accuracy of the line generation method. The accuracy of the unidimensional spectral method depends on the maximum frequency at which the spectrum is truncated ( $\Omega$ ), and the number of harmonics ( $M$ ). The model covariance function is given by

$$C_1(\zeta) = 2 \frac{\Omega}{M} \sum_{k=1}^M S_1(\omega_k) \cos(\omega_k \zeta) \tag{37}$$

where  $\omega_k = (k - 1/2)\Omega/M$ . Again take the case of a unit variance two-dimensional exponential type covariance function. The corresponding unidimensional covariance function  $C_1(\cdot)$  is given by (36) having spectral density function given by (26) and Table 1. Figure 8 compares the theoretical unidimensional covariance function (solid curves) to the covariance function given by the spectral model (37). In Figure 8a the number of harmonics is kept equal to  $M = 50$ , while the maximum frequency  $\Omega$  increases from  $\Omega = 10b$  to  $\Omega = 40b$ . For small  $\Omega$  (small  $\Delta\omega = \Omega/M$ ) there is very good accuracy for large distances but some error for small distances (especially in the variance). Figure 8b compares cases with  $\Omega = 40b$  and  $M$  varying from 50 to 100. While the accuracy for  $M = 50$  is poor at large distances, the accuracy shown for  $M = 100$  improves rapidly. This last case ( $\Omega = 40b, M = 100$ ) accurately preserves both the variance and the correlation at distances large relative to the correlation length  $b^{-1}$ .

EXAMPLES OF TWO-DIMENSIONAL SIMULATIONS

*Contour Map of a Two-Dimensional Realization*

Figure 1 is a contour map of a realization of a random field generated via the TBM, using the spectral method for line simulation. The field has a mean of five and an exponential covariance function with unit variance ( $\sigma^2 = 1$ ). The dimen-

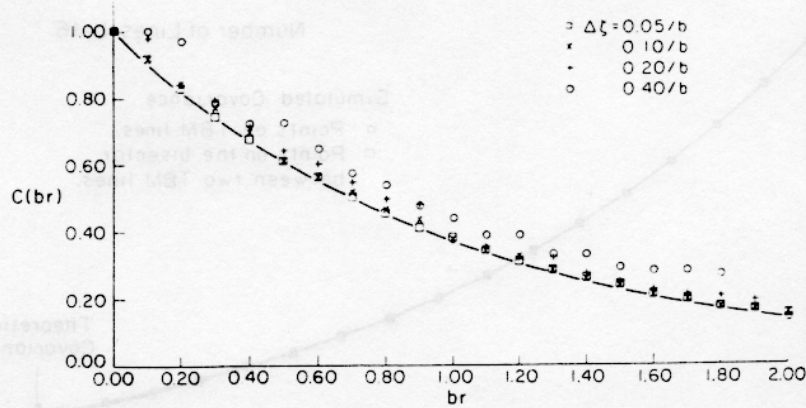


Fig. 7. Error in the covariance due to the discretization (band width) of the turning bands lines. The theoretical two-dimensional exponential covariance function has unit variance and is described by the solid line; the simulations are represented by the symbols.  $L = 16$ .

sions of the area are  $x_{max} = 1.2b^{-1}$  by  $y_{max} = 0.96b^{-1}$ . Simulations take place at discrete locations, shown as the grided points in the figure. There are  $NP = 8000$  of these simulation points arranged in 100 columns and 80 rows, with uniform spacing  $\Delta x = \Delta y = 0.012b^{-1}$  (see Figure 9). Sixteen evenly spaced TBM lines ( $L = 16$ ), a band width of  $\Delta\zeta = 0.012b^{-1}$ ,  $M = 100$  harmonics along the lines, and a maximum frequency  $\Omega = 40b$  are used in the generator.

How well does a single realization like this one, or a sequence of many realizations, preserve the statistics of the underlying random field? Some further examples will demonstrate.

*Spatial Statistics of One Realization*

The process generated by the TBM should preserve the spatial statistics. In order to demonstrate this, consider an

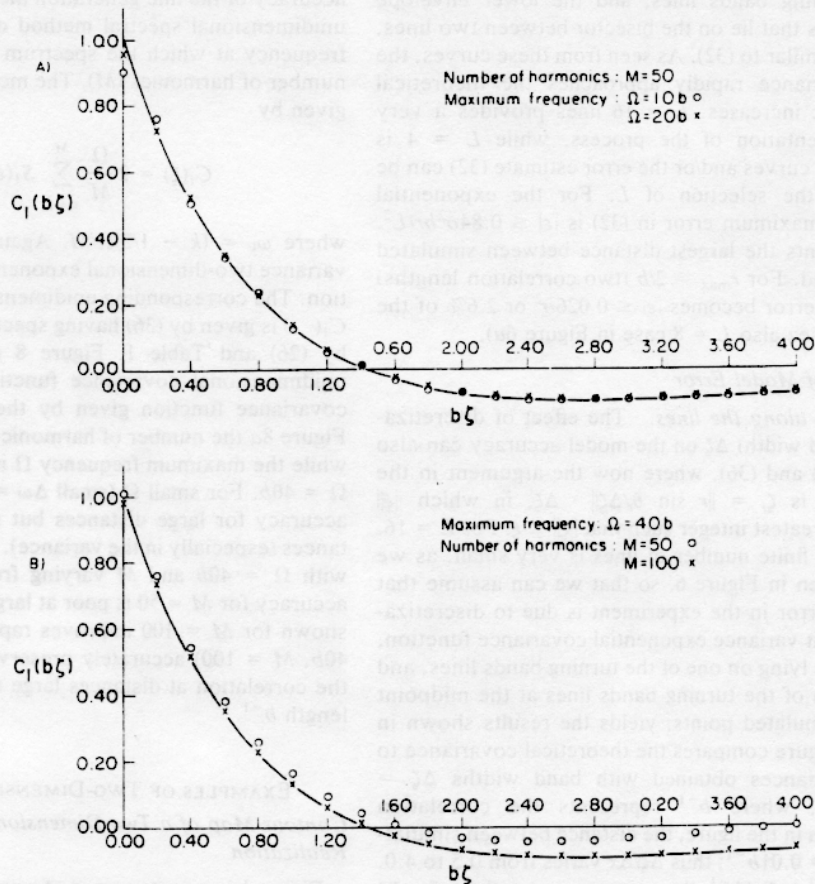


Fig. 8. Error in the unidimensional covariance  $C_1(b\zeta)$  (36) due to approximations in the spectral method of line generation (37). The theoretical two-dimensional covariance is exponential and has unit variance. (a) Constant number of harmonics  $M$  with varying frequency range  $\Omega$ . (b) Constant frequency range  $\Omega$  with varying number of harmonics  $M$ .



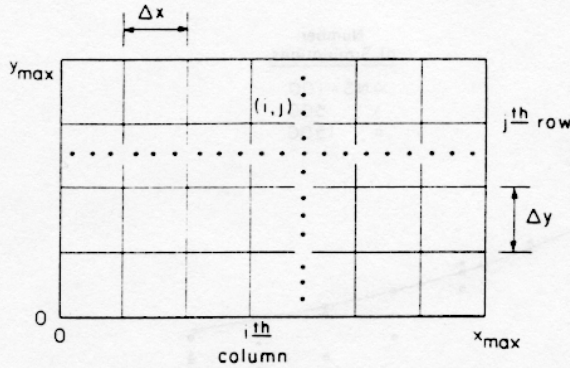


Fig. 9. Schematic of two-dimensional grided field.

example field of square area with dimensions  $x_{max} = y_{max} = 9b^{-1}$ , described by a zero mean process with unit variance ( $\sigma^2 = 1$ ) and exponential covariance function. Simulate a realization of this field at  $NP = 8100$  points, arranged in a  $90 \times 90$  grid, using the TBM with  $L = 16$ ,  $\Delta\zeta = 0.09b^{-1}$ ,  $M = 100$ , and  $\Omega = 40b$  ( $\Delta\omega = 0.40b$ ). The statistics of a typical realization of this process, as obtained by spatial averaging, are shown in Table 2. The theoretical and simulated statistics are very close, even though there is a simulation estimation error due to the finite dimensions of the field and the finite number of points generated. This estimation error tends to zero as the dimensions of the field and the number of simulated points increase. Thus the simulated field preserves the spatial statistics. Since the ensemble statistics are also preserved (see below) this example demonstrates the ergodicity property.

*Ensemble Statistics of a Finite Number of Realizations*

Another experiment describes the convergence of the ensemble statistics for a large number of simulations toward the theoretical values. In this experiment, realizations of a two-dimensional field with zero mean, unit variance, and exponential covariance are generated at a series of  $NP = 11$  points lying along an arbitrary straight line in  $R^2$ . Each point is separated by the other by a distance  $\Delta x = 0.20b^{-1}$ . The TBM parameters in the generation are  $L = 16$ ,  $\Delta\zeta = 0.075b^{-1}$ ,  $M = 75$ , and  $\Omega = 30b$  ( $\Delta\omega = 0.4b$ ). Figure 10 presents the results of this experiment for  $NS = 100, 500$ , and  $1500$  consecutive simulation trials, demonstrating that ensemble statistics converge toward the theoretical values for large  $NS$ . The observable differences between simulated and theoretical values is mostly due to estimation error because of the finite number of simulations. The standard

deviation of the estimated mean  $\bar{z}_{NS}$  from  $NS$  independent simulations is  $\sigma_{z,NS} = \sigma/(NS)^{1/2}$ . Since  $\sigma = 1$  here, we have  $\sigma_{z,100} = 0.10$ ,  $\sigma_{z,500} = 0.045$ , and  $\sigma_{z,1500} = 0.026$ . These values can be used to define confidence limits for Figure 10, demonstrating that the observable deviation in each case is mostly due to estimation error.

COMPARISON TO OTHER SIMULATION METHODS

The efficiency of the turning bands method (TBM) becomes more evident when compared to other simulation methods in terms of accuracy and cost. The well-known two-dimensional spectral methods of *Shinozuka and Jan* [1972] and *Mejia and Rodriguez-Iturbe* [1974] are the major state of the art techniques. Afterwards, the TBM is compared to matrix methods, in particular the nearest neighbor approach and the matrix decomposition approach of *Wilson* [1979].

*Accuracy Compared to Spectral Methods*

With lines evenly spaced at prespecified directions on the unit circle, the TBM is ergodic, even for a finite number of lines. Furthermore, the error in the simulated covariance function due to a finite number of lines rapidly approaches zero as  $1/L^2$  as the number of lines increases.

The multidimensional spectral method (SMR) proposed by *Mejia and Rodriguez-Iturbe* [1974] and earlier by *Shinozuka* [1971] has several problems. For a finite number of harmonics  $H$  this method possesses no error in the estimation of the covariance function from an ensemble average, but there is a large expected error in the spatial average estimation. Thus the process is ergodic only as  $H \rightarrow \infty$ . *Mejia and Rodriguez-Iturbe* give the standard deviation of their unbiased spatially averaged covariance function as

$$\sigma_{e,MR}(r) = \{\text{Var}[C_s(r)]\}^{1/2} = \frac{1}{\sqrt{H}} \left( \frac{\sigma^2 C(2r) - 2C^2(r) + \sigma^4}{2} \right)^{1/2} \quad (38)$$

where  $C_s$  is the covariance simulated with the spectral method, demonstrating that it converges to the theoretical function slowly as  $1/\sqrt{H}$ . Let us compare this to the TBM. Its error standard deviation ((34), (35)) converges much faster to zero, as  $1/L^2$ . For an exponential two-dimensional theoretical covariance,  $C(r) = \sigma^2 \exp(-br)$ , the SMR error standard deviation (38) becomes  $\sigma_{e,MR} = (\sigma^2/\sqrt{2H})[1 - \exp(-2br)]^{1/2}$ . The equivalent TBM error standard deviation, say (34), is  $\sigma_{e,TB} = 0.36\sigma^2 br/L^2$ . Given a TBM example with  $L = 8$ ,  $\Delta\zeta = 0.1b$ ,  $\Omega = 40$ , and  $M = 100$ , the

TABLE 2. Spatial Statistics of a Single Realization

	Mean	Variance (Standard Deviation)	Covariance for lag $j$ , $C(j)$ where $r = 2j\Delta x = j/5b$									
			$C(1)$	$C(2)$	$C(3)$	$C(4)$	$C(5)$	$C(6)$	$C(7)$	$C(8)$	$C(9)$	$C(10)$
Spatial statistics from the realization	-0.0059	1.021 (1.010)	0.826	0.704	0.569	0.458	0.366	0.282	0.214	0.173	0.139	0.111
Theoretical statistics	0.0000	1.000 (1.000)	0.813	0.670	0.549	0.449	0.368	0.301	0.247	0.202	0.165	0.135

$L = 16$ ,  $\Delta\zeta = 0.09b^{-1}$ ,  $M = 100$ ,  $\Omega = 40b$ ,  $NP = 8100$ .

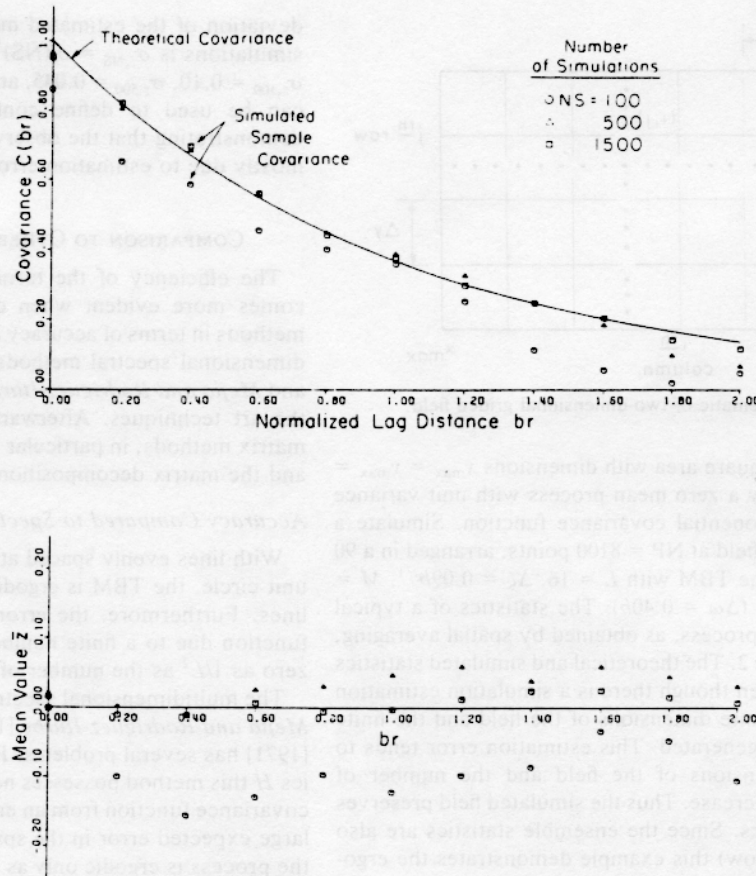


Fig. 10. Theoretical and simulated sample covariances, as a function of ensemble sample size  $NS$ . The theoretical two-dimensional covariance function has unit variance and is described by the solid line.  $N = 11$  points were sampled.  $L = 16$ ,  $\Delta\zeta = 0.075b^{-1}$ ,  $M = 75$ ,  $\Omega = 30b$ . See Mantoglou and Wilson [1981] for additional experimental statistics.

only appreciable TBM model error is due to the finite number of lines (see Figures 6, 7, and 8). Then for equivalent errors of the two methods we have  $\sigma_{f,MR} \approx \sigma_{f,TB}$ . If the maximum distances across example fields are  $br = 0.5, 2.0, 5.0$ , it would take, respectively,  $H = 40,000, 3900$ , and 630 SMR harmonics to provide the same 'accuracy' as our eight turning bands lines! Changing the criteria for comparing these techniques may slightly modify these numbers [see Mantoglou and Wilson, 1981], but there is no doubt that the TBM converges much faster than the SMR.

The spectral method (SMS) of Shinozuka and Jan [1972] has the advantage over the method of Mejia and Rodriguez-Iturbe [1974] in that it is ergodic and converges rapidly as the number of harmonics increases, but the cost of the method increases rapidly, too. Taking into account that for the TBM unidimensional process in our example above we used  $M = 100$  harmonics, then for the same level of accuracy in the generation of the SMS two-dimensional process we should approximately (depending on the shape of the spectral density function) use  $H = H_1 \times H_2 = (100)^2$  SMS harmonics. From the symmetry properties of the two-dimensional spectral density function in the isotropic case this number could be reduced to  $H = (1/2)(100/2)^2 = 1250$  SMS harmonics, which is still a large number (see cost comparisons later). A more comprehensive calculation of these errors is necessary for a thorough comparison.

#### Cost Compared to Spectral Methods

The cost of the TBM,  $c_{TB}$ , is essentially proportional to the number of lines ( $L$ ), to the number of harmonics used in the unidimensional process ( $M$ ), to the main diagonal (or largest) distance across the simulated field ( $D$ ), and to the number of simulations performed ( $NS$ ), while it is inversely proportional to the discretization length along the lines ( $\Delta\zeta$ ), or  $c_{TB} \propto NS \times L \times M \times D/\Delta\zeta$ . For a given accuracy (constant  $L, M, \Delta\zeta$ ) the cost is proportional to the distance  $D$  across the field. This means that increasing the size of an area, while maintaining its shape and the spacing of the generated points, increases the cost of the TBM as the square root of the number of points generated ( $NP$ ):  $c_{TB} \propto \sqrt{NP}$ . The cost is also shape dependent: the TBM is most efficient for circular shapes.

The cost of the SMR and SMS techniques are proportional to the number of simulations ( $NS$ ) performed, to the number of harmonics ( $H$ ), and to the number of points generated ( $NP$ ), or  $c_{SM} \propto NS \times H \times NP \propto NP$ , while the TBM cost is only proportional to the square root of the number of points. This property makes the TBM much less expensive to apply than other methods, particularly when a large number of points are generated.

We present here an example comparing relative cost (measured in CPU time on an IBM 370/168 CMS system) of

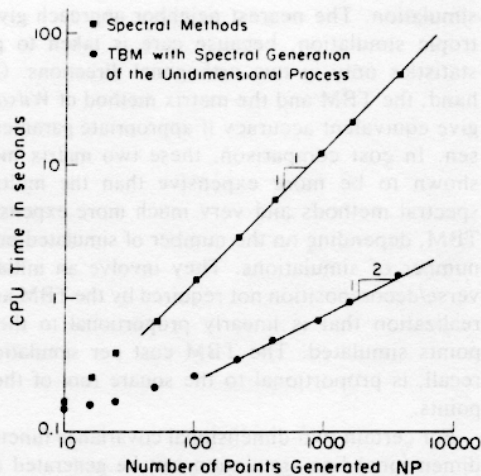


Fig. 11. Comparison of the turning bands method (TBM) to spectral methods (SMR) in terms of cost, for a square area with a uniform grid of simulated points. The grid spacing is maintained constant while the area is expanded uniformly.

the different methods. For the same number of harmonics the SMS of *Shinozuka and Jan* [1972] has approximately the same cost as the SMR of *Mejia and Rodriguez-Iturbe* [1974]. We have chosen to test the cost for these spectral methods with  $H = 200$  harmonics versus a turning bands model with spectral line generation and  $L = 8$ ,  $M = 100$ ,  $\Omega = 40b$ ,  $\Delta\omega = 0.40b$ , and  $\Delta\zeta = 0.025 b^{-1}$ . The maximum distance across the field is  $D = 2b^{-1}$ . Following the discussion above, the accuracy of the TBM with these parameters is much better than that of the spectral methods.

The example employs a square grid (see Figure 9) with a fixed grid spacing  $\Delta x = \Delta y = 0.025b^{-1}$ . The grid spacing is maintained while the size of the area and the number of simulated points (NP) are increased incrementally:  $NP = [(x_{\max}/\Delta x) + 1]^2$ , where  $x_{\max} = y_{\max}$  is the side dimension of the square area. For each size grid, one realization of the field at NP points is generated using both the SMR and the TBM, observing CPU time. Figure 11 is a plot of CPU time as a function of the number of simulated points. The figure clearly demonstrates that the TBM, with spectral line generating process, is superior in terms of cost to the SMR and, by inference, to SMS. The figure also confirms that  $c_{TB} \propto \sqrt{NP}$  while  $c_{SM} \propto NP$ , as noted above. For  $NP = 400$  points in a  $20 \times 20$  grid, two dimensional spectral methods cost 12 times more. For  $NP = 2500$  points in a  $50 \times 50$  grid, the cost is 32 times higher!

*Mantoglou and Wilson* [1982a] propose a TBM with generation of the line simulation as a moving average process. By using this model we can gain a slight reduction in cost in comparison to the TBM with the spectral line process [see *Mantoglou and Wilson*, 1982a, Figure 7], but we can only model certain special two dimensional covariance functions. The TBM with spectral line process does not possess this restriction.

#### Comparison to Matrix Techniques

The most common matrix technique used to simulate hydrologic random fields is the nearest neighbor approach [e.g., *Bartlett*, 1975; *Smith and Freeze*, 1979]. In two dimensions this method discretizes the domain into a series of blocks and relates the value of the field in each block to its

four neighbors. Most of the recent hydrologic applications neglect to note that the block statistics are spatially averaged and thus fail to account for 'variance-covariance reduction,' which may become very significant (see discussion by *Dettinger and Wilson* [1982] and *Wilson and Mantoglou* [1982]). The major drawback with the nearest neighbor approach is that it provides a very particular correlation structure that is not directly related to any isotropic two-dimensional covariance function, nor can the autoregressive parameters of the model be easily estimated directly from field data. Thus various specified correlation structures are only crudely simulated. This approach yields an anisotropic field because care is taken to preserve the correlation structure only in two orthogonal directions. The correlation preserved at, say, an angle of  $\pi/4$  between the two major axes of the nearest neighbor grid is different from that preserved on the major axes. Because of these features the nearest neighbor approach is clearly inferior to the TBM in terms of accuracy and versatility.

*Wilson* [1979] suggested a simple matrix decomposition technique that preserves the statistics of any proper two-dimensional covariance function and accounts of the implied spatial averaging. This technique results in simulated fields that are ergodic, stationary, and isotropic (for isotropic covariance functions). It is of the same order of accuracy as the TBM for appropriate choice of the parameters.

The computational effort for both the nearest neighbor approach and the matrix decomposition approach are similar. The former requires first the inversion of an  $NP \times NP$  matrix to begin the simulation, where NP is the number of simulated points. By carefully taking advantage of the structure of this matrix, the cost for doing this can be reduced from the order of  $NP^3$  to  $NP^2$ . Analogously, the matrix decomposition approach requires one matrix decomposition of an  $NP \times NP$  covariance matrix with a cost on the order of  $NP^2$  to  $NP^3$ . For each realization produced, both procedures require a vector-matrix product with a cost of the order of NP. Thus the total cost for these methods is roughly  $c_{MM} = \alpha \times NP^3 + \beta \times NS \times NP$ , where NS is the number of simulations and  $\alpha$  and  $\beta$  are proportionality parameters. Given that the cost of the TBM is  $c_{TBM} = \gamma \times NS \times \sqrt{NP}$ , we infer that for a large number of points the TBM is much more efficient than the matrix decomposition or nearest neighbor approach. However, for only a few points the matrix methods may be less expensive. The number of points NP at which the TBM is preferred depends on the parameters  $\alpha$ ,  $\beta$ , and  $\gamma$ , and these in turn depend on the problem being simulated. A more direct comparison of these methods in application is certainly called for, and we encourage it.

The matrix methods are most appropriate for spatially averaged processes. Fortunately, the TBM can be extended to the direct simulation of these processes, by using a direction dependent unidimensional spectral method [*Mantoglou and Wilson*, 1981, 1982a; *Wilson and Mantoglou*, 1982], and the significant cost savings implied above are preserved.

#### SUMMARY AND CONCLUSIONS

We have presented the turning bands method (TBM) for the simulation of random fields, with special emphasis on two-dimensional stationary processes. The originality of TBM, first introduced by *Matheron* [1973], is that it trans-



forms the two- or three-dimensional simulation problem to a unidimensional one. The results of several one-dimensional simulations are added to generate the point value of the two- or three-dimensional field. In hydrology the TBM can be applied to the synthetic generation of conditioned or unconditioned fields of spatially variable hydrologic properties and inputs, such as hydraulic conductivity and aquifer recharge in groundwater hydrology and land slope, soil hydraulic properties, and precipitation in surface water hydrology. It also has applications in the fields of mining, ocean engineering, geotechnical engineering, etc.

For two-dimensional fields, we have introduced an equation relating the spectral density function of the unidimensional TBM line process to the radial spectral density function of the two-dimensional field. The line process can now be generated using a spectral method, permitting for the first time the simulation of two-dimensional random fields with the TBM having any properly posed covariance function. Although all of our examples were drawn using the two-dimensional exponential covariance function, (26) and Table 1 readily provide the appropriate expressions for the simulation of other covariance functions.

We investigated the accuracy of the TBM. The generated process is ergodic for a finite number of lines, if the lines are evenly spaced on the unit circle with prespecified directions and if the line generation method itself is ergodic. The simulated covariance quickly converges to the theoretical covariance with  $1/L^2$ , where  $L$  is the number of lines, as demonstrated both theoretically and by example. We found that  $L = 4-16$  lines are sufficient for almost all applications in two dimensions. We also developed guidelines for choosing model parameters during the design of simulation runs, in order to limit model error. With judicious parameter selection, the TBM is both inexpensive and accurate.

Using the ergodic unidimensional spectral method of Rice [1954] and Shinozuka and Jan [1972] for generation along the lines, we produced a series of examples to illustrate typical two-dimensional simulation results. In one of these examples we calculated the spatial sample statistics of one realization: within the limits of simulation estimation error caused by a finite sample size, the generated field preserved the theoretical spatial statistics. In another example we ran a sequence of experiments involving up to 1500 realizations, demonstrating the convergence of ensemble statistics toward the theoretical values as the number of simulations increases. Taken together, these two examples demonstrate that both ensemble and spatial averages preserve the theoretical statistics, thus confirming by experiment the ergodicity of the TBM.

We compared the TBM to two-dimensional spectral generation models. The TBM is considerably less expensive than either of the two-dimensional spectral methods of Mejia and Rodriguez-Iturbe [1974] and Shinozuka and Jan [1972] for the same level of accuracy. The TBM has a cost proportional to the square root of the number of points simulated, while the other methods have a cost linearly proportional to the number of points. This relative cost was amply demonstrated through a case study of grid generation.

We also compared the TBM to two-dimensional matrix methods. The TBM is superior to the nearest neighbor approach because it is not restricted to particular covariance structures typifying this approach. The TBM is more accurate and, for an isotropic field, it provides an isotropic

simulation. The nearest neighbor approach gives an anisotropic simulation, because care is taken to preserve the statistics only in two orthogonal directions. On the other hand, the TBM and the matrix method of Wilson [1979] can give equivalent accuracy if appropriate parameters are chosen. In cost comparison, these two matrix methods were shown to be more expensive than the multidimensional spectral methods and very much more expensive than the TBM, depending on the number of simulated points and the number of simulations. They involve an initial matrix inverse/decomposition not required by the TBM and a cost per realization that is linearly proportional to the number of points simulated. The TBM cost per simulation, one will recall, is proportional to the square root of the number of points.

For certain two-dimensional covariance functions the unidimensional line simulation can be generated as a moving average process (see Chiles [1977] and Mantoglou and Wilson [1981, 1982a] for two different special cases of this type). This can be slightly less expensive than the case of unidimensional spectral method [Mantoglou and Wilson, 1982a] but is more restrictive.

Using the unidimensional spectral method for line simulation, the turning bands method can easily be extended to the direct generation of stationary anisotropic random fields [Mantoglou and Wilson, 1981, 1982b] and consequently to the direct generation of spatially averaged stationary random fields [Mantoglou and Wilson, 1981; Wilson and Mantoglou, 1982], such as occur in the finite difference cells or finite elements of numerical Monte Carlo simulation models of hydrologic processes [see, e.g., Wilson and Mantoglou, 1982; Dettinger and Wilson, 1982]. Multiple parameter correlation can be added to account for correlation between, say, hydraulic conductivity and porosity in a groundwater transport model input. The temporal correlation of aquifer recharge or rainfall over a watershed can also be accounted for. Nonstationary random fields of the intrinsic type can be simulated by the TBM [Matheron, 1973], using polynomial generalized covariance representations and a Wiener (Brownian motion) process along the lines [Orfeuil, 1972; Mantoglou and Wilson, 1981]. Finally, the simulations can be conditioned on observations, using standard techniques [e.g., David, 1973; Journel and Huijbregts, 1978; Delhomme, 1979].

All of the simulation examples presented in this paper were performed on an IBM 360/168 using a code written by the authors, which also includes the capability of synthetically generating anisotropic, areal averaged, and intrinsic random fields. Taken together with a recently completed estimation program using generalized kriging [Kafritsas and Bras, 1981], up to date estimation and simulation techniques are now readily available in the United States.

#### APPENDIX

In this appendix we calculate a first-order approximation of the error in the simulated two-dimensional covariance function obtained with the turning bands method (TBM) due to a finite number of lines ( $i = 1, L$ ). The TBM lines are assumed to be evenly spaced on the unit circle, with prespecified angles ( $\theta_i$ ). The error  $\varepsilon$  in the covariance function is defined by  $\varepsilon = C_i(r) - C(r)$ , where  $C(r)$  is the theoretical covariance function given by the first expression

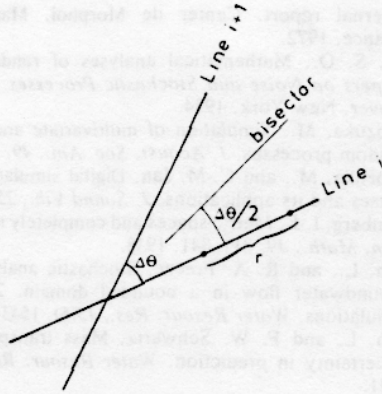


Fig. 12. Location of the bisector between two TBM lines,  $i$  and  $i + 1$ .

in (15) and  $C_s(r)$  is the simulated covariance function given by (9) with  $\mathbf{h} \cdot \mathbf{u}_i = r \sin \theta_i$ . Thus the error  $\epsilon$  is given by (31).

Let us assume that  $L$  is an even number. Also, for the moment, assume that the points separated by distance  $r$  where the covariance function is calculated lie on one of the turning bands lines. Let  $i$  and  $i + 1$  be two adjacent TBM lines, as shown in Figure 12. The constant angle formed between these lines is  $\Delta\theta = \pi/L$ . By substituting  $g(\theta) = C_1(r \sin \theta)$  in the theoretical covariance (15), we get:

$$C(r) = \frac{1}{\pi} \int_0^\pi g(\theta) d\theta = \frac{1}{\pi} \sum_{i=1}^L \int_{\theta_i}^{\theta_i + \Delta\theta} g(\theta) d\theta = \frac{1}{\pi} \sum_{i=1}^L I_i \tag{A1}$$

where  $\theta_1 = 0$  and  $\theta_i = (i - 1)\Delta\theta$ . The symbol  $I_i$  represents the integral of  $g(\theta)$  between  $\theta_i$  and  $\theta_i + \Delta\theta$ . Expanding  $g(\theta)$  in a Taylor series with center  $\theta_i$ , this integral becomes

$$I_i = g(\theta_i)\Delta\theta + g'(\theta_i) \frac{\Delta\theta^2}{2} + g''(\theta_i) \frac{\Delta\theta^3}{6} - 0[\Delta\theta^4]$$

where the prime indicates a  $\theta$  derivative, so that the theoretical covariance function has the following approximation for small  $\Delta\theta$ :

$$C(r) \cong \frac{1}{\pi} \left\{ \sum_{i=1}^L C_1(r \sin \theta_i) \Delta\theta + \frac{1}{2} \sum_{i=1}^L [C_1(r \sin \theta_i)]' \Delta\theta^2 + \frac{1}{6} \sum_{i=1}^L [C_1(r \sin \theta_i)]'' \Delta\theta^3 \right\} \tag{A2}$$

Since  $\Delta\theta = \pi/L$ , the first term of this series cancels out the first term of the right-hand side of the error term (31), leaving

$$\epsilon \cong -\frac{1}{2L} \sum_{i=1}^L [C_1(r \sin \theta_i)]' \Delta\theta - \frac{\pi}{6L^2} \sum_{i=1}^L [C_1(r \sin \theta_i)]'' \Delta\theta \tag{A3}$$

Redefining the function  $g(\theta_i) = [C_1(r \sin \theta_i)]'$ , the first term of this error estimate can also be approximated using the integral series approximation (A1) and a Taylor series expansion:

$$\int_0^\pi [C_1(r \sin \theta)]' d\theta \cong \sum_{i=1}^L [C_1(r \sin \theta_i)]' \Delta\theta$$

$$+ \frac{1}{2} \sum_{i=1}^L [C_1(r \sin \theta_i)]'' \Delta\theta^2 + 0[\Delta\theta^3] \tag{A4}$$

where higher order terms have been neglected. Since the left-hand side is zero, this can be rearranged to give

$$\sum_{i=1}^L [C_1(r \sin \theta_i)]' \Delta\theta \cong -\frac{\pi}{2L} \sum_{i=1}^L [C_1(r \sin \theta_i)]'' \Delta\theta \tag{A5}$$

By substituting into (A3), we get the estimated covariance error:

$$\begin{aligned} \epsilon &\cong \frac{\pi}{12L^2} \sum_{i=1}^L [C_1(r \sin \theta_i)]'' \Delta\theta \cong \frac{\pi}{12L^2} \int_0^\pi [C_1(r \sin \theta)]'' d\theta \\ &= \frac{\pi}{12L^2} \frac{dC_1(r \sin \theta)}{d\theta} \Big|_{\theta=0}^\pi = \frac{\pi\sigma^2 Kbr}{6L^2} \end{aligned} \tag{A6}$$

where we have approximated the summation by an integral, which we have evaluated at its limits. The coefficient  $K$ , in which  $b^{-1}$  represents correlation length, is found by the chain rule:

$$K = -\frac{1}{\sigma^2 b} \frac{dC_1(\zeta)}{d\zeta} \Big|_{\zeta=0} \tag{A7}$$

It depends on the type of covariance function.

This estimation for the error assumes that there are an even number of evenly spaced lines on the unit circle and that the points where the covariance function is calculated lie on one of the turning bands lines. If these points lie on the bisector between two adjacent lines, we found the error to be given by

$$\epsilon \cong \frac{-\pi\sigma^2 Kbr}{12L^2} \tag{A8}$$

Thus the error lies within the bounds given by (32).

These derivations are based on small angles  $\Delta\theta$ . Through examination of the Taylor series we can also show that these error approximations imply small distance  $r$  relative to the correlation length  $b^{-1}$  and are not valid when  $r$  becomes large.

*Acknowledgments.* The authors wish to express their thanks to Rafael Bras of the Massachusetts Institute of Technology (MIT), whose comments during the early stages of this work were extremely valuable. The work was partially supported by a grant to the MIT/Cairo University Technology Planning Program from the U.S. Agency for International Development.

REFERENCES

Abramowitz, M., and I. A. Stegun, *Handbook of Mathematical Functions, Appl. Math. Ser. 55*, National Bureau of Standards, Washington, D. C., 1964.  
 Bailey, W. N., *Generalized Hypergeometric Series*, Cambridge University Press, London, 1953.  
 Bartlett, M. S., *The Statistical Analysis of Spatial Pattern*, Chapman and Hall, London, 1975.  
 Chiles, J. P., *Geostatistique des phenomenes non stationnaires*, These de Docteur-Ingénieur, Univ. de Nancy, Nancy, France, 1977.  
 David, M., *Geostatistical Ore Reserve Estimation*, Elsevier, New York, 1977.  
 Delhomme, J. P., Kriging in the hydrosociences, *Advan. Water Resour.*, 1(5), 251-266, 1978.  
 Delhomme, J. P., Spatial variability and uncertainty in groundwater flow parameters: A geostatistical approach, *Water Resour. Res.*, 15(2), 269-280, 1979.

- Dettinger, M. D., and J. L. Wilson. First-order analysis of uncertainty in numerical models of groundwater flow. 1. Mathematical development. *Water Resour. Res.*, 17(1), 149-161, 1981.
- Dettinger, M. D., and J. L. Wilson. First-order analysis of uncertainty in numerical models of groundwater flow. 2. Examples. submitted to *Water Resour. Res.*, 1982.
- Freeze, R. A., A stochastic-conceptual analysis of rainfall-runoff processes on a hillslope, *Water Resour. Res.*, 16(2), 391-408, 1980.
- Gradshteyn, I. S., and I. M. Ryzhik. *Table of Integrals, Series and Products*. Academic, New York, 1980.
- Journel, A. B., and Ch. J. Huijbregts. *Mining Geostatistics*. Academic, New York, 1978.
- Kafritsas, J., and R. L. Bras. The practice of kriging. *Tech. Rep. 263*. Ralph M. Parsons Lab., Dep. of Civ. Eng., Mass. Inst. of Technol., Cambridge, Mass., 1981.
- Mantoglou, A., and J. L. Wilson. Simulation of random fields with the turning bands method. *Tech. Rep. 264*. Ralph M. Parsons Lab., Dep. of Civ. Eng., Mass. Inst. of Technol., Cambridge, Mass., 1981.
- Mantoglou, A., and J. L. Wilson. Simulation of two dimensional random fields via the turning bands method, using line generation by moving averages. submitted to *J. Hydrol.*, 1982a.
- Mantoglou, A., and J. L. Wilson. Simulation of anisotropic random fields by the turning bands method, submitted to *Water Resour. Res.*, 1982b.
- Matern, B., Spatial variation. *Medd. Statens Skogsforskninginst. Swed.*, 49(5), 1960.
- Matheron, G., The intrinsic random functions and their applications. *Advan. Appl. Prob.*, 5, 439-468, 1973.
- Mejia, J., and I. Rodriguez-Iturbe. On the synthesis of random fields from the spectrum: An application to the generation of hydrologic spatial processes. *Water Resour. Res.*, 10(4), 705-711, 1974.
- Orfeuil, J. P., Simulation du Wiener-Lévy et de ses intégrals. internal report, Center de Morphol. Math., Fontainebleau, France, 1972.
- Rice, S. O., Mathematical analyses of random noise. *Selected Papers on Noise and Stochastic Processes*, edited by N. Wax, Dover, New York, 1954.
- Shinozuka, M., Simulation of multivariate and multi-dimensional random processes. *J. Acoust. Soc. Am.*, 49, 357-367, 1971.
- Shinozuka, M., and C.-M. Jan. Digital simulation of random processes and its applications. *J. Sound Vib.*, 25(1), 111-128, 1972.
- Shoenberg, I. J., Metric spaces and completely monotone functions. *Ann. Math.*, 39, 811-841, 1938.
- Smith, L., and R. A. Freeze. Stochastic analysis of steady state groundwater flow in a bounded domain. 2. Two-dimensional simulations. *Water Resour. Res.*, 15(6), 1543-1559, 1979.
- Smith, L. and F. W. Schwartz. Mass transport. 2. Analysis of uncertainty in prediction. *Water Resour. Res.*, 17(2), 351-369, 1981.
- Veneziano, D. A., Random processes for engineering applications. course notes, Dep. of Civ. Eng., Mass. Inst. of Technol., Cambridge, Mass., 1978.
- Wilson, C. B., J. B. Valdes, and I. Rodriguez-Iturbe. On the influence of the spatial distribution of rainfall on storm runoff. *Water Resour. Res.*, 15(2), 321-328, 1979.
- Wilson, J. L., The synthetic generation of areal averages of random field, paper presented at Socorro Workshop on Stochastic Methods in Subsurface Hydrology, New Mex. Tech., Socorro, New Mex., April 26-27, 1979.
- Wilson, J. L., and A. Mantoglou, Areal average stochastic processes and their simulation, submitted to *Water Resour. Res.*, 1982.

(Received October 5, 1981;

revised March 23, 1982;

accepted April 5, 1982.)

## **Appendix B**

### **TUBA version 2.11d Computer Code Listing**





PROGRAM TUBA

```

C-----C
C Main program module for TUBA, Version 2.11d C
C-----C
C TUBA revision history ... C
C ver 2.0 - originally released in July, 1990 (see manual) C
C ver 2.10 - allows reproduction of individual fields from an ensemble; C
C generation onto irregularly-space finite difference grids C
C ver 2.11 - never released (essentially a beta test version) C
C Much more memory efficient; mean and variance for GCF; C
C computation progress flag; change in the default line C
C discretization length and bug fix (see page 116) C
C to be consistent with guidelines given in Chapter 3. C
C Option for multiple realizations into 1 or many files C
C ver 2.11a - same as ver 2.11 but with improved "status reporting" C
C over beta version 2.11 C
C ver 2.11b - shortens line process discretization length for the C
C spectral-based covariance models (by factor of .01) C
C ver 2.11c - has option to match the desired mean and variance stats C
C exactly and a bug fix for use with mask file option C
C ver 2.11d - has option to allow for a nugget effect in the field(s) C
C Not asked for desired mean and variance for GC model C
C unless scaling to match desired stats option invoked. C
C
C NOTE: because of changes in line-discretization length parameters C
C described above for updates 2.11 and 2.11b, THE MEAN AND C
C VARIANCE STATISTICS WILL NOT MATCH THE EXAMPLES PRINTED IN C
C THE MANUAL, however, the examples are intended only for illus- C
C trative purposes on how to properly apply the code. C
C-----C
C For Documentation, see: C
C
C Zimmerman, D. A. and John L. Wilson. July, 1990 C
C "Description of and User's Manual for TUBA: C
C A Computer Code for Generating Two-dimensional Random Fields C
C via the Turning Bands Method" C
C
C Published by C
C
C SETA, Inc. C
C 27 Ponderosa Drive, Suite 100 C
C Cedar Crest, New Mexico 87153 USA C
C http://www.setainc.com C
C-----C
C TUBA version 2.0 was developed in the C
C Geoscience Department (Hydrology Program) C
C New Mexico Institute of Mining and Technology, C
C Socorro, New Mexico 87801 USA C
C-----C
C This version has WRITE(IT,*) statements after every input query be- C
C cause Lahey Fortran apparently does not add the <cr> automatically. C
C-----C
C INCLUDE FILE FOR TUBA VERSION 2.11d C
C COMMON /TBAPAR/ ICOVF,IPAA,LINES,FMAX,NHAR,NMAX,UN,FX,FY, C
C 1 XO,YO,TBMX,KS,IP,NX,NY,XMAX,YMAX,DX,DY,NXY, AM,AN,AV,CLX,CLY, C
C 1 IDFP,IURN, DS,UD,KD,NR,CK,FM,FA, A1,A3,A5,KT,DT,SG,B0,B1,B2, C
C 1 NF,IPF,SAJ,IULP,MSK,IMSEX C
C-----C
C LOGICAL UNIT IDENTIFIERS C
C IN = standard input - terminal ( generally, this will be unit 5 ) C
C IT = standard output - terminal ( generally, this will be unit 6 ) C
C IL = listing file unit C
C IO = output data unit C
C L1 = used for reading (x,y) points, mask file data, then for storing C
C the (x,y) generation points in direct access file for gridded fields C
C L2 = used for areal average processes - stores line process data for C
C each line in direct access file (to reduce memory requirements) C
C-----C
C PARAMETER (IN=5,IT=6,IL=7,IO=20,L1=21,L2=22,LGTH=199000) C
C COMPLEX C(LGTH/2) C
C REAL A(LGTH) C
C EQUIVALENCE (A,C) C
C
C INCLUDE 'tuba211d.inc' C
C COMMON /PGPARS/ PCL,IPG

```

```

COMMON /ADRSES/LXY,LPP,LPA,LZ1,LZZ,LZM,LSS,LCC,LFF,LTT,LDZ,
1      LS1,LS2,LC1,LC2

C      READ INPUT AND OUTPUT PARAMETERS
      CALL RDINPT(IN,IT,IL,L1,A(1),A(1),MODEL,NLINE,NSIM)

C      CALCULATE INTERNAL PARAMETERS
      CALL INTPAR(A(1))

C      CREATE THE <NAME>.INP CARD FILE
      CALL LSTINP(IL)

C      CALCULATE "ADDRESSES" OF ARRAY POINTERS
      CALL ADDRES(IT,IL,LGTH)

C      CALCULATE (X,Y) POINT PROJECTIONS ONTO THE TBM LINES
      CALL CALXYP(IT,L1,A(LXY),A(LZM),A(LSS),A(LCC),A(LPP))

C      CALCULATION OF LINE PROCESS ARRAY DATA
      CALL CALINP(IT,L2,A(LPA),A(LSS),A(LCC),A(LS1),A(LC1),A(LFF))

C      BEGIN SIMULATING THE RANDOM FIELD(S)
      DO 20 ISIM=1,NSIM
        WRITE(IT,*)
        IF(NSIM.GT.1) CALL PROGSS(' SIMULATION NUMBER ... ',
1          IT,ISIM,NSIM,5)
        PCLSAV = PCL
        IPGSAV = IPG
        CALL RESEED(ISIM,NUSEED)
        DO 10 L=1,NLINE
          IF(MODEL.LE.3) CALL SPCTRL(L,L2,A(LPA),A(LZ1),C(LDZ/2+1),
1            A(LS1),A(LC1),A(LS2),A(LC2))
          IF(MODEL.EQ.4) CALL MOVAVG(A(LTT),A(LZ1),A(LFF))
          IF(MODEL.EQ.5) CALL WNRLVY(A(LZ1))
          CALL PROJCT(L,IT,L1,A(LXY),A(LPP),A(LSS),A(LCC),A(LZZ),
1            A(LZM),A(LZ1))
10      CONTINUE
        CALL OUTPUT(IT,IL,IO,ISIM,NSIM,A(LXY),A(LZZ),A(LZM),NUSEED)
        PCL = PCLSAV
20      CONTINUE

      STOP
      END

BLOCK DATA
-----
C
C      INITIALIZE DATA FOR VARIABLES IN LABELED COMMON STATEMENTS

      INCLUDE 'tuba211d.inc'

Comt COMMON /SEEDS/ needed by URNITMB
COMMON /SEEDS/ ML,MM,MK,L,M,K

Comt COMMON /ADRSES/ needed by ADDRES, MAIN
COMMON /ADRSES/LXY,LPP,LPA,LZ1,LZZ,LZM,LSS,LCC,LFF,LTT,LDZ,
1      LS1,LS2,LC1,LC2

Comt COMMON /IRSGRD/ needed by DEFPAR, FLDPAR
COMMON /IRSGRD/ GXMIN,GYMIN

Comt DATA L, M, K and ML, MM, MK needed by URNITMB
DATA L, M, K /089347405, 301467177, 240420681/
DATA ML,MM,MK /65539, 33554433, 36243609/

Comt DATA LS1,LS2 ... needed by ADRSES
DATA LS1,LS2,LC1,LC2 /1,1,1,1/, MSK /0/

Comt DATA FM,FA,AM .. needed by INTPAR
DATA FM,FA,AM,AN,AV,CLX,CLY /1.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0/

Comt DATA GXMIN,GYMIN needed by FLDPAR
DATA GXMIN,GYMIN /1.E+15,1.E+15/

      END

```

```

SUBROUTINE ADDRES(IT,IL,LGTH)
C-----
C   CALCULATE ADDRESSES OF ARRAY POINTERS

C   LXY,LPP,LPA, ETC. ARE THE "ADDRESSES" OF THE XY,PP,PA ARRAYS IN
C   THE ONE DIMENSIONAL ARRAY A. IF SIMULATING AT ARBITRARY (X,Y)
C   LOCATIONS (KS=1), THESE COORDINATES (NXY OF THEM) ARE STORED
C   AT THE BEGINNING OF ARRAY A. IF A MASK FILE IS USED, IT IS ALSO
C   (MUTUALLY EXCLUSIVE OPTIONS) STORED AT THE BEGINNING OF ARRAY A.

      INCLUDE 'tuba211d.inc'
      CHARACTER BITES*10
      COMMON /ADRSES/LXY,LPP,LPA,LZ1,LZZ,LZM,LSS,LCC,LFF,LTT,LDZ,
1         LS1,LS2,LC1,LC2
Comt SEE BLOCK DATA MODULE
Comt DATA  LS1,LS2,LC1,LC2 /1,1,1,1/

      IGS = 0
      KSS = MIN(2,KS)
      IF(KS.GT.1) NXY = NX*NY
      IF(KS.EQ.3) IGS = NX+NY
      LXY = 1
      LZM = 1 + IGS
      LPP = (2-KSS)*(2*NXY) + 1 + MIN(MSK,1)*NXY + IGS
      LPA = LPP + 2*(KSS-1)*NX
      LZ1 = LPA + NHAR
      LZZ = LZ1 + MAX(NMAX,NHAR)
      LSS = LZZ + NXY
      LCC = LSS + LINES
      LFF = LCC + LINES
      LTT = LFF + KD
      LDZ = LTT + NR
      LRQ = LDZ + 2*NHAR

C   THESE ARRAYS ARE FOR SHINOZUKA AND JAN METHOD
      IF(ICOVF.LE.3 .AND. ISAJ.EQ.1) THEN
          LS1 = LTT + NR
          LC1 = LS1 + NHAR
          LS2 = LC1 + NHAR
          LC2 = LS2 + NHAR
          LRQ = LC2 + NHAR
      END IF

C   DIRECT ACCESS FILES USE AN ADDITIONAL 8 BYTES PER RECORD
C   BXY = BYTES FOR (X,Y) DATA; BAA = BYTES FOR AREAL AVERAGE DATA
      BXY = ( 4*(2*NXY) + 8*NY )
      BAA = ( 4*NHAR*LINES + 8*LINES ) * (IPAA-1)
      FDS = ( BXY + BAA ) * 1.E-06
      BITES = ' Megabytes'
      IF(FDS.LT.1) THEN
          BITES = ' Kilobytes'
          FDS = 1000 * FDS
      END IF
      WRITE(IT,10) LGTH,LRQ,LGTH-LRQ,FDS,BITES
      WRITE(IL,10) LGTH,LRQ,LGTH-LRQ,FDS,BITES
10  FORMAT(/' Number Of Elements Allocated In A Array =',I9,
1     /' Total Storage Required For Computations =',I9,
1     /' No Of Elements In Excess of Requirements =',I9,/' Free',
1     /' Disk Space Needed For Computations =',F9.3,A,/)

      IF(LRQ.LT.0 .AND. IDFP.EQ.2) WRITE(IT,15)
15  FORMAT(' ***** err, integer overflow - you may be specifying',
1     /' the Turning Band line parameters improperly.',
2     /' Recheck Turning Band parameter input values')
C   The following should NEVER occur
      IF(LRQ.LT.0 .AND. IDFP.EQ.1) WRITE(IT,16)
16  FORMAT(' ***** err, integer overflow - contact code author',
1     /' send email message to: dazimme@somnet.sandia.gov')
      IF(LRQ.GT.LGTH) WRITE(IT,20)
20  FORMAT(' ***** err, Insufficient Storage (Dimension of A Array)',
1     /' Increase Parameter LGTH in main program')

      IF(LRQ.GT.LGTH) STOP

      RETURN
      END

```

```

SUBROUTINE CALINP(IT,L2,PA,SS,CC,S1,C1,FF)
C-----
C   CALCULATION OF ARRAY DATA NEEDED FOR LINE PROCESS GENERATION

REAL    PA(*), SS(*), CC(*), S1(*), C1(*), FF(*)
INCLUDE 'tuba211d.inc'
COMMON  /PGPARS/ PCL,IPG
CHARACTER LPMA*47,LPSM*43,LPTB*43,LPSP*47
DATA    LPMA /' Calculating Line Process Data ... (MA Process)'/
DATA    LPSM /' Calculating Line Process Data ... Harmonic'/
DATA    LPTB /' Calculating Line Process Data ... Line No'/
DATA    LPSP /' Calculating Line Process Data ... ( Spectral )'/

C   NO ARRAY DATA NEEDED FOR NON-STATIONARY GENERALIZED COVARIANCE MODELS
C   GC MODEL LINE PROCESS PARAMETERS CALCULATED IN SUBROUTINE INTPAR
IF(ICOVF.EQ.5) RETURN
Comt WRITE(IT,*)'Calculating Line Process Data ... (GC Models)

C   MOVING AVERAGE GENERATION OF THE LINE PROCESS
IF(IULP.EQ.2) THEN
C   USER-DEFINED MOVING AVERAGE ALGORITHM GOES HERE
C   THE FF ARRAY CONTAINS THE MOVING AVERAGE WEIGHTS
C   SEE SECTIONS 2.4 AND 5.4 OF THE USER'S MANUAL
Comt WRITE(IT,*)' Calculating Line Process Data ... (MA Process)
IF(IPF.GE.2) WRITE(IT,'(A)') LPMA
CK = 1.0
RETURN
END IF

C   MOVING AVERAGE GENERATION OF THE LINE PROCESS (TELIS COVARIANCE)
IF(ICOVF.EQ.4) THEN
Comt WRITE(IT,*)' Calculating Line Process Data ... (MA Process)
IF(IPF.GE.2) WRITE(IT,'(A)') LPMA
DO 10 K=1,KD
XT = DS*FLOAT(K-1)
FF(K) = (1.-XT)*EXP(-XT)
10 CONTINUE
CX = 1.-EXP(-2.*DS)
CK = SQRT(12.*CX*CX/(CX-DS*EXP(-2.*DS)))
RETURN
END IF

C   SPECTRAL GENERATION OF THE LINE PROCESS (SAJ AND FFT METHODS)
AX = FX/CLX
AY = FY/CLY
DOM = FMAX/FLOAT(NHAR)
DLM = 0.1*DOM
IF(IPF.EQ.1) WRITE(IT,'(A)') LPSP
IF(IPAA.EQ.2) GO TO 33

C   line generation for POINT processes ...
DO 20 M=1,NHAR
Comt CALL PROGSS(' Calculating Line Process Data ... Harmonic',
IF(IPF.GE.2) CALL PROGSS(LPSM,IT,M,NHAR,20)
OM = (FLOAT(M)-0.5)*DOM
SPEC = SPDF(OM,ICOVF)*DOM
C   This if block pertains only to the Shinozuka and Jan method
IF(ISAJ.EQ.1) THEN
OMM = OM + URN55()*DLM
C1(M) = COS(OMM*UN)
S1(M) = SIN(OMM*UN)
SPEC = 2.0*SQRT(SPEC)
END IF
PA(M) = SPEC
20 CONTINUE
RETURN

C   line generation for AREAL AVERAGE processes ...
33 LREC = 4*NHAR
OPEN(UNIT=L2,STATUS='SCRATCH',ACCESS='DIRECT',RECL=LREC,
1 FORM='UNFORMATTED')
DO 40 L=1,LINES
IF(IPF.GE.2) CALL PROGSS(LPTB,IT,L,LINES,10)
comt IF(IPF.EQ.2) CALL PROGSS(LPTB,IT,L,LINES,10)
comt IF(IPF.EQ.3) CALL PROGSS(' Turning Band Line ...',IT,L,LINES,10)
comt PCLSAV = PCL

```

```

comt   IPGSAV = IPG
       DO 30 M=1,NHAR
Comt   CALL PROGSS(' Calculating Line Process Data ... Harmonic',
comt   IF(IPF*IPGSAV.EQ.3) CALL PROGSS(LPSM,IT,M,NHAR,20)
       OM = (FLOAT(M)-0.5)*DOM
       SPEC = SPDF(OM,ICOVF)*DOM
C      This if block pertains only to the Shinozuka and Jan method
       IF(ISAJ.EQ.1) THEN
           OMM = OM + URN55()*DLM
           C1(M) = COS(OMM*UN)
           S1(M) = SIN(OMM*UN)
           SPEC = 2.0*SQRT(SPEC)
       END IF
       AC = CC(L)
       AS = SS(L)
       IF(ICOVF.EQ.0) AASD = WTUSR(OM,AC,AS,AX,AY)*SPEC
       IF(ICOVF.EQ.1) AASD = WTEXP(OM,AC,AS,AX,AY)*SPEC
       IF(ISAJ.EQ.1) AASD = 2.*SQRT(AASD)
       PA(M) = AASD
30     CONTINUE
       WRITE(UNIT=L2,REC=L) (PA(M),M=1,NHAR)
comt   PCL = PCLSAV
40     CONTINUE

       RETURN
       END

```

SUBROUTINE CALXYC(K,GS,X,Y)

```

C-----
C      CALCULATE (x,y) COORDINATES FOR GRIDDED OUTPUT

       REAL    GS(*)
       INCLUDE 'tuba211d.inc'

C      KS = 1  OUTPUT AT SPECIFIED (X,Y) LOCATIONS
C      KS = 2  OUTPUT ONTO A BLOCK OR POINT CENTERED REGULARLY SPACED GRID
C      KS = 3  OUTPUT ONTO A BLOCK OR POINT CENTERED IRREGULARLY SPACED GRID
C      IP = 1  POINT CENTERED GRID,   IP = 2  BLOCK CENTERED GRID

C      DECODE I AND J INDICES FROM SINGLE INDEX REFERENCE
       J = K/NX + 1
       IF(MOD(K,NX).EQ.0) J = J - 1
       I = K - (J-1)*NX
       V = 1.0/FLOAT(IP)
       IF(KS.EQ.2) X = (I-V)*DX
       IF(KS.EQ.2) Y = (J-V)*DY
       IF(KS.EQ.3) THEN
           X = 0
           Y = 0
           DO 10 II=1,I-1
10          X = X + GS(II)

           X = X + (1-V)*GS(I)
           DO 20 JJ=1,J-1
20          Y = Y + GS(NX+JJ)
           Y = Y + (1-V)*GS(NX+J)
       END IF

       RETURN
       END

```

SUBROUTINE CALXYP(IT,IU,GS,ZM,SS,CC,PP)

```

C-----
C      CALCULATE (X,Y) POINT PROJECTIONS ONTO THE TBM LINES

       PARAMETER (PI=3.141592654)
       REAL      CC(*), SS(*), GS(*), ZM(*), PP(2,*)
       INCLUDE 'tuba211d.inc'
       CHARACTER PPCS*43
       DATA     PPCS /' Calculating Projection Points ... Point No'/'

```

```

C   CALCULATE SINES AND COSINES OF TBM LINE ANGLES
      DTHA = PI/FLOAT(LINES)
      TNOT = URN55() * 2.0*PI
      DO 10 L=1,LINES
            THETA = FLOAT(L)*DTHA + TNOT
            CC(L) = COS(THETA)
            SS(L) = SIN(THETA)
10    CONTINUE

C   NORMALIZED PROJECTION POINTS FOR KS=1 OBTAINED IN SUBROUTINE PROJCT
      IF(KS.EQ.1) THEN
            WRITE(IT,*)'Projection Points Not Calculated -> Read as Input'
            WRITE(IT,*)'Projection Points = (x,y) Field Generation Points'
            IF(IULP.EQ.2 .OR. ICOVF.GE.4) WRITE(IT,*)
            RETURN
      END IF

C   CALCULATE AND STORE THE NORMALIZED (X,Y) PROJECTION POINTS FOR GRIDS
      LREC = 2*(4*NX)
      OPEN(UNIT=IU,STATUS='SCRATCH',ACCESS='DIRECT',RECL=LREC,
1      FORM='UNFORMATTED')
      IF(IPF.EQ.1) WRITE(IT,*)'Calculating Projection Points ...'
      DO 40 I=1,NY
            DO 30 J=1,NX
                  K = (I-1)*NX + J
Comt    CALL PROGSS(' Calculating Projection Points ... Point No',
                  IF(IPF.EQ.2) CALL PROGSS(PPCS,IT,K,NXY,20)
                  IF(IPF.EQ.3) CALL PROGSS(PPCS,IT,K,NXY, 5)
                  IF(MSK.GT.0 .AND. ZM(K).EQ.0) GO TO 30
                  CALL CALXYC(K,GS,X,Y)
                  PP(1,J) = X
                  PP(2,J) = Y
30      CONTINUE
            WRITE(UNIT=IU,REC=I) ((PP(K,J),K=1,2),J=1,NX)
40     CONTINUE
      WRITE(IT,*)

      RETURN
      END

```

SUBROUTINE COMENT(ID,CMT,NC)

```

C-----
C   RETURN COMMENT FOR THE <NAME>.INP FILE

C   TUBA CREATES A CARD FILE "ON THE FLY" (ie. WHILE IT IS BEING RUN
C   INTERACTIVELY). THE CARD FILE PROVIDES A RECORD OF WHAT OPTIONS AND
C   PARAMETERS WERE USED AND LISTS THE SAMPLE STATISTICS OF EACH OUTPUT
C   FIELD. THE CARD FILE CAN ALSO BE USED FOR BATCH PROCESSING.

CHARACTER  COMTS(37)*42, CMT*(*)
INTEGER    NCHRS(37)
DATA  COMTS( 1) /'1=(x,y) Locations, 2=Even Grid, 3=Uneven  '/
DATA  COMTS( 2) /'Input Filename for (x,y) Locations      '/
DATA  COMTS( 3) /'1=Point Centered, 2=Block Centered     '/
DATA  COMTS( 4) /'Maximum X and Y Field Dimensions        '/
DATA  COMTS( 5) /'Number of Nodes-X and Nodes-Y          '/
DATA  COMTS( 6) /'1=Normal, 2=exp(X), 3=10**(X)          '/
DATA  COMTS( 7) /'0=User,1=Exp,2=Gauss,3=Besl,4=Telis,5=GC '/
DATA  COMTS( 8) /'1=Point Process, 2=Areal Average Process '/
DATA  COMTS( 9) /'X and Y Dimensions of Averaging Area    '/
DATA  COMTS(10) /'Desired Mean, Nugget and Sill           '/
DATA  COMTS(11) /'X and Y Direction Correlation Lengths   '/
DATA  COMTS(12) /'Generalized Covariance Model Coefficients '/
DATA  COMTS(13) /'Line Process by: 1=Spectral, 2=Moving Avg '/
DATA  COMTS(14) /'1=Default TBM Parameters, 2=Enter Manually'/
DATA  COMTS(15) /'Number of Turning Band Lines            '/
DATA  COMTS(16) /'TBM Line Discretization Distance       '/
DATA  COMTS(17) /'Nbr of Harmonics for Discretizing Spectrum'/
DATA  COMTS(18) /'Max Frequency for Truncation of Spectrum '/
DATA  COMTS(19) /'Discretization Distance for MA Process  '/
DATA  COMTS(20) /'Discretization Distance for Weiner Process'/
DATA  COMTS(21) /'Field ORIGIN Relative to TBM Origin    '/
DATA  COMTS(22) /'Output Data Filename                   '/
DATA  COMTS(23) /'1=Output Only Z, 2=Output X,Y, and Z   '/

```

```

DATA COMTS(24) /'1=Unformatted, 2=Formatted Output      '//
DATA COMTS(25) /'Output Format for Writing Data to Disk  '//
DATA COMTS(26) /'1=Single Write Statement, 2=Line at a Time'//
DATA COMTS(27) /'1=First Row to Last, 2=Last Row to First '//
DATA COMTS(28) /'1=Marsaglia URNG, 2=Machine Indep URNG  '//
DATA COMTS(29) /'Seed(s) for Random Number Generator    '//
DATA COMTS(30) /'Number of Realizations to be Simulated  '//
DATA COMTS(31) /'Maximum Turning Band Line Length      '//
DATA COMTS(32) /'Mask Filename                          '//
DATA COMTS(33) /'Input Filename for grid-block widths   '//
DATA COMTS(34) /'1=Single file output, 2=Multiple files '//
DATA COMTS(35) /'1=Minimal, 2=Med, 3=Frequent screen output'//
DATA COMTS(36) /'0=do not scale, 1=match T-stats exactly '//
DATA COMTS(37) /'Mean, Nugget, Sill for Gen Cov Model   '//

```

```

DATA NCHRS      / 40, 34, 35, 32, 29, 29, 40, 40, 36, 29, 37,
1              / 41, 41, 42, 28, 32, 42, 40, 38, 42, 35, 20,
1              / 36, 33, 38, 42, 40, 38, 35, 38, 32, 13, 36,
1              / 38, 42, 39, 36/

```

```

CMT = COMTS(ID)
NC = NCHRS(ID)

```

```

RETURN
END

```

```

SUBROUTINE COVPAR(IN,IT,IL,MODEL)

```

```

C-----
C QUERY FOR COVARIANCE PARAMETERS OF THE RANDOM FIELD

INCLUDE 'tuba211d.inc'

WRITE(IT,10)
10 FORMAT(/' ++++++++ COVARIANCE PARAMETERS +++++++'/)

WRITE(IT,*)'Select Type Of Covariance Model:'
WRITE(IT,*)'(0) - User Specified'
WRITE(IT,*)'(1) - Exponential Model'
WRITE(IT,*)'(2) - Gaussian Covariance'
WRITE(IT,*)'(3) - Bessel Type Covariance'
WRITE(IT,*)'(4) - Telis Covariance Function'
WRITE(IT,*)'(5) - Generalized Covariance Model'
WRITE(IT,*)
CALL RDINTG(IN,IT,IL,ICOVF,1,7)
MODEL = ICOVF

IPAA = 1
C AREAL AVERAGE PROCESS DATA FOR USER DEFINED OR EXPONENTIAL MODELS
IF(ICOVF.LE.1) THEN
  WRITE(IT,*)'(1) - Point Process'
  WRITE(IT,*)'(2) - Areal Average Process'
  WRITE(IT,*)
  CALL RDINTG(IN,IT,IL,IPAA,1,8)
  IF(IPAA.EQ.2) THEN
    WRITE(IT,*)'Enter X And Y Dimensions Of Averaging Rectangle'
    WRITE(IT,*)
    CALL RDREAL(IN,IT,IL,FX,2,9)
  END IF
END IF

IF(ICOVF.EQ.5) THEN
  WRITE(IT,*)'Enter Gen. Covariance Parameters A1,A3,A5'
  WRITE(IT,*)' K(r) = A1*r + A3*r**3 + A5*r**5'
  WRITE(IT,*)' A1,A5.GE.0, A3.GE.-(10/3)*SQRT(A1*A5) '
  WRITE(IT,*)
  CALL RDREAL(IN,IT,IL,A1,3,12)
  RETURN
END IF

WRITE(IT,*)'Enter Mean And Variance Parameters: If You Will Have'
WRITE(IT,*)'The Field(s) Exponentiated, Enter The Desired Mean &'
WRITE(IT,*)'Variance BEFORE Exponentiation (Variance=Nugget+Sill)'
WRITE(IT,*)
WRITE(IT,*)'Enter The Mean, Nugget And Sill For Covariance Model'

```

```

WRITE(IT,*)
CALL RDREAL(IN,IT,IL,AM,3,10)
IF(AN.LT.0 .OR. AV.LT.0) THEN
  STOP '***** ERR, nugget and sill must be > or = 0'
END IF

WRITE(IT,*)'Enter The X and Y Direction Correlation Lengths'
WRITE(IT,*)'  Make These Equal For Isotropic Fields '
WRITE(IT,*)
CALL RDREAL(IN,IT,IL,CLX,2,11)

RETURN
END

```

```

SUBROUTINE DEFPAR(NLINE,XY)

```

```

C-----
C  CALCULATE DEFAULT TURNING BAND PARAMETERS

PARAMETER (PI=3.141592654)
REAL      DLK(4),XY(*)
INTEGER   NHR(4)
INCLUDE 'tuba211d.inc'
COMMON /IRSGRD/ GXMIN,GYMIN

DATA      NHR /2048, 1024, 4096, 0 /
DATA      DLK /0.05, 0.10, 0.025, 0./

C  CALCULATE DEFAULT TBM ORIGIN AND MAXIMUM TBM LINE LENGTH
CALL ORGMAX(XY)

C  SET DEFAULT NUMBER OF TBM LINES (BOTH NEEDED BECAUSE OF COMMON)
C  THERE IS NO LONGER A "DEFAULT" NUMBER OF TBM LINES ...
Comt     LINES = 16
Comt     NLINE = 16

C  SPECTRAL AND MOVING AVERAGE METHOD LINE PARAMETERS
IF(ICOVF.LE.4) THEN
  ISAJ = 0
  NHAR = NHR(ICOVF)
  UN = 0.0625
  DS = 0.05*AMIN1(C LX,CLY)
END IF

C  BLOCK OR CELL SPACING FOR GRIDDED OUTPUT
IF(KS.EQ.2) THEN
  K = 2 - IP
  DX = XMAX / MAX(NX-K,1)
  DY = YMAX / MAX(NY-K,1)
ELSE IF(KS.EQ.3) THEN
  DX = GXMIN
  DY = GYMIN
END IF

C  GENERALIZED COVARIANCE MODEL DEFAULT LINE DISCRETIZATION DISTANCE
C  DX,DY SET IN ORGMAX FOR THE CASE OF KS=1
IF(ICOVF.EQ.5) THEN
  UN = AMIN1(DX,DY)
  DT = 0.2*UN
END IF

C  UN = NORMALIZED LINE DISCRETIZATION DISTANCE = 2*PI/FMAX (FFT METHOD)
C  FOR STATIONARY MODELS, UN IS SET EQUAL TO 0.0625 (16 PTS/CORR LGTH)
C  IF(UN.GT.DX .OR. UN.GT.DY) THEN UN IS DECREASED APPROPRIATELY.
C  ALSO MAKE SURE MA PROCESS PARAMETER DS IS SMALL ENOUGH.
IF(KS.GT.1 .AND. ICOVF.LE.4) THEN
  CM = AMAX1(C LX,CLY)
  DC = AMIN1(C LX/16.,CLY/16.)
  DN = AMIN1(.99*DX/CLX,.99*DY/CLY,DC/CM)
  UN = AMIN1(UN,DN)
  DS = AMIN1(DS,UN*AMIN1(C LX,CLY)/10)
END IF

C  IF THE SPECTRAL METHOD IS USED .AND. UN IS DECREASED, THEN THE
C  FREQUENCY SPACING DELK MAY BE GREATER THAN THE ALLOWABLE MAXIMUM

```



```

C (SEE DLK IN DATA STATEMENT). WHEN THIS HAPPENS, NHAR IS INCREASED
C (BY A FACTOR OF 2 FOR THE FFT) TO OBTAIN A SMALLER DELK.

      FMAX = 2.*PI/UN
      IF(UN.LT.0.0625 .AND. ICOVF.LE.3) THEN
16      IF(FMAX/FLOAT(NHAR).GT.DLK(ICOVF)) THEN
          NHAR = 2*NHAR
          GO TO 16
      END IF
      END IF

      RETURN
      END

```

SUBROUTINE FFT(F,NPT,IFB)

```

C-----
C ONE DIMENSIONAL FAST FOURIER TRANSFORM ROUTINE (FORWARD AND INVERSE)
C
C THIS ROUTINE, MODIFIED BY D. A. (TONY) ZIMMERMAN AT NEW MEXICO TECH
C AND VERIFIED WITH IMSL ROUTINES, WAS TAKEN FROM PAGE 108 OF:
C
C RAFAEL .C GONZALEZ AND PAUL WINTZ, 1987.
C 'DIGITAL IMAGE PROCESSING'
C ADDISON-WESLEY PUBLISHING COMPANY
C
C F = COMPLEX SEQUENCE TO BE TRANSFORMED (INPUT)
C F = COMPLEX TRANSFORMED ARRAY ON OUTPUT
C NPT = NUMBER POINTS IN F TO BE TRANSFORMED
C IFB = -1 FOR FORWARD TRANSFORM ( EXP(-i*2PIux/NPT) )
C IFB = +1 FOR INVERSE TRANSFORM ( EXP(+i*2PIux/NPT) )
C-----
C NOTE: THE INPUT SEQUENCE MUST BE OF LENGTH EQUAL TO 2**N FOR SOME N
C-----

```

```

      PARAMETER (PI=3.141592654)
      COMPLEX F(*),U,W,T

      IF(IFB.GT.0) THEN
10      DO 10 K=1,NPT
          F(K) = CONJG(F(K))
      END IF

      LN = ALOG(FLOAT(NPT))/ALOG(2.0)
      N = 2**LN
      NV2 = N/2
      NM1 = N-1

      J = 1
      DO 3 I=1,NM1
          IF(I.GE.J) GO TO 1
          T = F(J)
          F(J) = F(I)
          F(I) = T
1      K = NV2
2      IF(K.GE.J) GO TO 3
          J = J-K
          K = K/2
          GO TO 2
3      J = J+K

      DO 5 L=1,LN
          LE = 2**L
          LE1 = LE/2
          U = CMPLX(1.0,0.0)
          A = PI/LE1
          W = CMPLX(COS(A),-SIN(A))
          DO 5 J=1,LE1
              DO 4 I=J,N,LE
                  IP = I+LE1
                  T = F(IP)*U
                  F(IP) = F(I)-T
4                  F(I) = F(I)+T
5                  U = U*W

```

```

        IF(IFB.GT.0) THEN
          DO 20 K=1,NPT
            F(K) = CONJG(F(K))
          END IF

          RETURN
        END

```

```

SUBROUTINE FFTGEN(L,L2,PA,Z1,DZ)

```

```

C-----
C   GENERATION OF THE LINE PROCESS VIA FAST FOURIER TRANSFORM

```

```

      REAL    PA(*),Z1(*)
      COMPLEX DZ(*),i
      INCLUDE 'tuba211d.inc'

```

```

C   THE IMAGINARY PART YIELDS AN INDEPENDENT REALIZATION
      IF(MOD(L,2).EQ.0) GO TO 30

```

```

      i = (0.,1.)
      IF(IPAA.EQ.2) READ(UNIT=L2,REC=L) (PA(M),M=1,NHAR)

```

```

      DO 10 M=1,NHAR
        DELF = PA(M)
        SQDF = SQRT(6.*DELF)
        A = SQDF * URN55()
        B = SQDF * URN55()
        DZ(M) = A - i*B
      10 CONTINUE

```

```

      CALL FFT(DZ,NHAR,-1)

```

```

      DO 20 M=1,NHAR
        Z1(M) = 2.0*REAL(DZ(M))
      20

```

```

      RETURN

```

```

      30 DO 40 M=1,NHAR
        Z1(M) = 2.0*AIMAG(DZ(M))
      40

```

```

      RETURN
      END

```

```

SUBROUTINE FILPAR(IN,IT,IL)

```

```

C-----
C   QUERY FOR OUTPUT FILE PARAMETERS

```

```

      CHARACTER    FMT*35,FNAM*35
      INCLUDE 'tuba211d.inc'
      COMMON /OTPTS1/ FMT,FNAM
      COMMON /OTPTS2/ IFO,IMO,IRO,ILN

```

```

      WRITE(IT,5)
      5 FORMAT(//' ++++++ OUTPUT FILE PARAMETERS ++++++')

```

```

      WRITE(IT,*)'Enter A Filename For The Output File(s)'
      WRITE(IT,*)
      CALL RDCHAR(IN,IT,IL,FNAM,22)

```

```

      IF(KS.EQ.1) THEN
        WRITE(IT,*)'(1) - Output Only The Field Values, Z'
        WRITE(IT,*)'(2) - Output The (X,Y) Locations And Z'
        WRITE(IT,*)
        CALL RDINTG(IN,IT,IL,IOF,1,23)
      END IF

```

```

      WRITE(IT,*)'(1) - Unformatted Output'
      WRITE(IT,*)'(2) - Formatted Output'
      WRITE(IT,*)
      CALL RDINTG(IN,IT,IL,IFM,1,24)

```

```

IF(IFM.EQ.2) THEN
  IF(KS.EQ.1) WRITE(IT,10) '(2F12.2,1PE12.5)''
  IF(KS.GT.1) WRITE(IT,10) '(10F12.5)''
10  FORMAT(' Enter Output Format, e.g., ',A)
  WRITE(IT,*)' (include the parentheses)'
  WRITE(IT,*)
  CALL RDCHAR(IN,IT,IL,FMT,25)
END IF

IF(KS.GT.1) THEN
  WRITE(IT,*)'(1) - Write Out Matrix With One WRITE Statement'
  WRITE(IT,*)'(2) - Write Out Matrix One Line (Row) At A Time'
  WRITE(IT,*)
  CALL RDINTG(IN,IT,IL,IMO,1,26)
  IMO = 2 - IMO
  IRO = 1
  IF(IMO.EQ.0 .AND. IFM.EQ.2) THEN
    WRITE(IT,*)' Output the Rows of the Matrix via ...'
    WRITE(IT,*)'(1) - First Row --> Last Row'
    WRITE(IT,*)'(2) - Last Row --> First Row'
    WRITE(IT,*)
    CALL RDINTG(IN,IT,IL,IRO,1,27)
  END IF
END IF

C  IOF AND IFO ARE PARAMETERS CONTROLLING OUTPUT FORMAT
  KSS = MIN(KS,2)
  IOF = MAX(IOF*(2-KSS),1)
  IFO = IOF*2 - IFM + 1

  RETURN
  END

```

```

SUBROUTINE FLDPAR(IN,IT,IL,IU,XY,GS,ZM)

```

```

C-----
C  QUERY FOR OUTPUT FIELD PARAMETERS

  CHARACTER  FMT*35,FNAM*35, DATAF*35, MASKF*35
  REAL       XY(2,*),GS(*),ZM(*)
  INCLUDE 'tuba211d.inc'
  COMMON /IRSGRD/ GXMIN,GYMIN
  COMMON /OTPTS1/ FMT,FNAM
  COMMON /OTPTS2/ IFO,IMO,IRO,ILN

  WRITE(IT,5)
5  FORMAT('//' ++++++++ OUTPUT FIELD PARAMETERS ++++++++ '/')

  WRITE(IT,*)'(1) - Simulate Only At Specified (x,y) Locations'
  WRITE(IT,*)'(2) - Simulate Onto A Regularly Spaced Grid'
  WRITE(IT,*)'(3) - Simulate Onto An Unevenly Spaced Grid'
  WRITE(IT,*)
  CALL RDINTG(IN,IT,IL,KS,1,1)

  IF(KS.EQ.1) THEN
    WRITE(IT,*)'Enter The Filename For Reading (X,Y) Locations'
    WRITE(IT,*)
    CALL RDCHAR(IN,IT,IL,DATAF,2)
    OPEN(UNIT=IU,FILE=DATAF,STATUS='OLD')
    I = 0
10  I = I + 1
    READ(IU,*,END=11) XY(1,I), XY(2,I)
    GO TO 10
11  NXY = I-1
    WRITE(IT,20) NXY
20  FORMAT(' >>>> ',I8,' Data Pairs Read')
    CLOSE(UNIT=IU)
  END IF

  IF(KS.GT.1) THEN
    WRITE(IT,*)'(1) - Point Centered Grid'
    WRITE(IT,*)'(2) - Block Centered Grid'
    WRITE(IT,*)
    CALL RDINTG(IN,IT,IL,IP,1,3)
  END IF

```

```

WRITE(IT,*)'Enter The Maximum X And Y Field Dimensions'
WRITE(IT,*)
CALL RDREAL(IN,IT,IL,XMAX,2,4)

WRITE(IT,*)'Enter The Number Of Nodes In The X And Y Directions'
WRITE(IT,*)
CALL RDINTG(IN,IT,IL,NX,2,5)

IF(KS.EQ.3) THEN
  WRITE(IT,*)'Enter The Filename For Reading Grid-Block Widths'
  WRITE(IT,*)
  CALL RDCHAR(IN,IT,IL,DATAF,33)
  OPEN(UNIT=IU,FILE=DATAF,STATUS='OLD')
  READ(IU,*) (GS(I),I=1,NX-2+IP)
  READ(IU,*) (GS(NX+I),I=1,NY-2+IP)
  CLOSE(UNIT=IU)
  DO 30 I=1,NX
30   IF(GS(I).LT.GXMIN) GXMIN = GS(I)
  DO 32 I=1,NY
32   IF(GS(NX+I).LT.GYMIN) GYMIN = GS(NX+I)
  END IF

  WRITE(IT,*)'Enter mask filename or type NONE or <cr>'
  WRITE(IT,*)
  CALL RDCHAR(IN,IT,IL,MASKF,32)
  IF(MASKF.EQ.' ') MASKF = 'none'
  NONE = INDEX(MASKF,'NONE') + INDEX(MASKF,'none')
  IF(NONE.EQ.0) THEN
    OPEN(UNIT=IU,FILE=MASKF,STATUS='OLD')
    READ(IU,*) (ZM(I),I=1,NX*NY)
    CLOSE(UNIT=IU)
    DO 40 I=1,NX*NY
40   IF(ZM(I).NE.0) MSK = MSK + 1
  END IF
  END IF

  WRITE(IT,*)'(1) - Generate a Field f(x) Whose pdf is Gaussian'
  WRITE(IT,*)'(2) - Generate a Lognormal Field  $K(x) = \exp(f(x))$ '
  WRITE(IT,*)'(3) - Generate a Lognormal Field  $K(x) = 10^{**}(f(x))$ '
  WRITE(IT,*)
  CALL RDINTG(IN,IT,IL,ILN,1,6)

  RETURN
  END

SUBROUTINE FSCALE(XY,ZZ,ZM,ILN,BAR,VAR,SSQ,PTS)
C-----
C DO FINAL SCALING, ADD IN THE MEAN, NUGGET & CALCULATE MEAN AND VARIANCE

REAL ZZ(*), ZM(*), XY(2,*)
INCLUDE 'tuba211d.inc'

SUM = 0.0
SSQ = 0.0
HNUG = SQRT(3.0*AN)
SDEV = SQRT(AV)
SQLN = SQRT(FLOAT(LINES))
DO 10 K=1,NXY
  IF(MSK.EQ.0 .OR. (MSK.GT.0 .AND. ZM(K).NE.0)) THEN
    ZZ(K) = SDEV*ZZ(K)/SQLN
    ZZ(K) = ZZ(K) + AM
    IF(AN.GT.0) ZZ(K) = ZZ(K) + URNAB(-HNUG,+HNUG)
    SUM = SUM + ZZ(K)
    SSQ = SSQ + ZZ(K)*ZZ(K)
  END IF
10 CONTINUE
PTS = FLOAT(MSK)
IF(PTS.EQ.0) PTS = FLOAT(NXY)
BAR = SUM / PTS
VAR =(SSQ-PTS*BAR*BAR) / (PTS-1.)

C FROM THIS POINT ON WE CAN SIMPLY CHECK IF ZZ(K).NE.0 (.NE.EXACT ZERO)
C RATHER THAN CHECKING THE MSK FLAG AND ZM ARRAY (IF MASK OPTION USED)

```

```

C      USE THE FORMULA ON PAGE 10 OF THE MANUAL (THE FORMULA THAT'S LABELED
C      "DO NOT USE") TO SCALE THE FIELD TO MATCH THE DESIRED MEAN & VARIANCE.
C      READ SECTION 2.1.2 OF MANUAL BEFORE INVOKING THIS OPTION.
C      BAR LEFT IN FORMULAS BELOW FOR CLARITY.  AN=NUGGET, AV=SILL
      IF(IMSEX.EQ.1) THEN
        DO 30 I=1,NXY
30       IF(ZZ(I).NE.0) ZZ(I) = ZZ(I) - BAR
          BAR = 0.
          SFAC = SQRT(AN+AV) / SQRT(VAR-BAR*BAR)
          DO 40 I=1,NXY
40       IF(ZZ(I).NE.0) ZZ(I) = SFAC * (ZZ(I)-BAR) + AM
          BAR = AM
          VAR = AN + AV
        END IF
      comt if you don't believe BAR=AM, VAR=AN+AV, uncomment the following lines
      comt SUM = 0.
      comt SSQ = 0.
      comt DO 50 K=1,NXY
      comt   IF(ZZ(K).NE.0) THEN
      comt     SUM = SUM + ZZ(K)
      comt     SSQ = SSQ + ZZ(K)*ZZ(K)
      comt   END IF
50     CONTINUE
      comt BAR = SUM / PTS
      comt VAR =(SSQ-PTS*BAR*BAR) / (PTS-1.)

      comt EXPONENTIATE THE FIELD IF REQUESTED ...
      IF(ILN.EQ.2) THEN
        DO 60 K=1,NXY
60       IF(ZZ(K).NE.0) ZZ(K) = EXP(ZZ(K))
      ELSE IF(ILN.EQ.3) THEN
        DO 70 K=1,NXY
70       IF(ZZ(K).NE.0) ZZ(K) = 10.**(ZZ(K))
      END IF

      RETURN
      END

```

#### SUBROUTINE INTPAR(XY)

```

C-----
C      CALCULATE INTERNAL PARAMETERS

      PARAMETER (PI=3.141592654)
      REAL      XY(*)
      INCLUDE 'tuba211d.inc'
Comt SEE BLOCK DATA MODULE
Comt DATA      FM,FA,AM,AV,CLX,CLY /1.0, 0.0, 0.0, 1.0, 1.0, 1.0/

C      NORMALIZE DISTANCE FROM TBM ORIGIN TO OUTPUT FIELD ORIGIN
      XO = XO / CLX
      YO = YO / CLY

C      DX AND DY DEPEND ON WHETHER GRID IS POINT OR BLOCK CENTERED
      IF(KS.EQ.2) THEN
        K = 2 - IP
        DX = XMAX / MAX(NX-K,1)
        DY = YMAX / MAX(NY-K,1)
      END IF

C      NORMALIZE LINE DISCRETIZATION DISTANCE (UN) FOR STATIONARY MODELS
      IF(IDFP.NE.1 .AND. ICOVF.LE.4) UN = UN / AMAX1(CLX,CLY)

C      NORMALIZE MAX DISCRETIZATION DISTANCE ALONG ANY TBM LINE, THEN
C      ESTIMATE NMAX = THE NUMBER OF POINTS ALONG THE LONGEST TBM LINE
      IF(ICOVF.LE.4) THEN
        TBMX = TBMX / AMIN1(CLX,CLY)
        NMAX = TBMX / UN + 1
      END IF

C      FOR FFT GENERATION ALGORITHM, LINE PROCESS LENGTH = UN*NHAR/2.
C      IF THAT IS LESS THAN TBMX, NHAR MUST BE INCREASED.

```

```

16 IF(ICOVF.LE.3 .AND. ISAJ.EQ.0) THEN
    IF(UN*NHAR/2.0 .LT. TBMX) THEN
        NHAR = 2*NHAR
        GO TO 16
    END IF
END IF

C   CALCULATE PARAMETERS NEEDED FOR THE MOVING AVERAGE PROCESS
IF(ICOVF.EQ.4 .OR. IULP.EQ.2) THEN
    DS = DS/AMIN1(C LX,CLY)
    UD = UN/DS
C   FOR USER DEFINED MA PROCESS, CLN MUST BE REPLACED WITH A NUMBER
C   REPRESENTING THE NBR CORR LGTHS THE MA WEIGHTING FCN IS NON-ZERO
Comt IF(ICOVF.EQ.0) KD = CLN/DS + 1
    IF(ICOVF.EQ.4) KD = 5.0/DS + 1
    NR = NMAX*UD + KD
    IF(DS.GT.UN) FM = UD
    IF(DS.GT.UN) FA = 0.5
END IF

C   GENERALIZED COVARIANCE PARAMETERS
IF(ICOVF.EQ.5) THEN
    NMAX = TBMX/UN + 1
    KT = AMAX1(UN/DT,1.)
    DT = UN / FLOAT(KT)
    SG = SQRT( 24.*DT )
    B0 = SQRT(A1*PI/2.)
    B2 = SQRT(A5*PI*15./16.)
    B1 = A3*PI* 3./4.
    B1 = SQRT(B1*B1 + 2.*B0*B2)
END IF

RETURN
END

```

SUBROUTINE LSTINP(IL)

```

C-----
C   LIST INPUT VALUES & INTERNAL PARAMETERS USED IN LINE PROCESS GENERATION
C
C   AS INPUT IS READ, IT IS WRITTEN (AND ANNOTATED) ON UNIT IL. NOW REWIND
C   IL, REREAD THOSE LINES AND STORE THEM IN THE "INTERNAL FILE" BUF. THEN
C   OPEN IL WITH THE DATA FILENAME AND EXTENSION ".INP" & DUMP BUF INTO IT.
C-----
    INTEGER          NC(32)
    CHARACTER        FMT*35, FNAM*35, LSTF*35, REC*80, BUF(32)*80
    INCLUDE 'tuba211d.inc'
    COMMON /OTPTS1/ FMT, FNAM
    COMMON /OTPTS2/ IFO, IMO, IRO, ILN

C   READ THE ANNOTATED INPUT PARAMETERS FROM SCRATCH FILE (UNIT IL)
C   AND WRITE THEM TO THE INTERNAL FILE "BUF"
    REWIND IL
    DO 10 K=1,32
        READ(IL,5,END=11) REC
    5   FORMAT(A)
        WRITE(BUF(K),5) REC
        NC(K) = NCHR(REC)
    10  CONTINUE
    11  NREC = K-1

C   OPEN THE <NAME>.INP LISTING FILE; REUSE LOGICAL UNIT IL AND
C   WRITE THE ANNOTATED INPUT PARAMETERS TO THIS LIST FILE
    CLOSE(UNIT=IL,STATUS='DELETE')
    IDOT = INDEX(FNAM, '.')
    IF(IDOT.EQ.0) IDOT = NCHR(FNAM) + 1
    LSTF = FNAM(1:IDOT-1) // '.inp'
    OPEN(UNIT=IL,FILE=LSTF,STATUS='UNKNOWN',FORM='FORMATTED')
    DO 15 K=1,NREC
    15  WRITE(IL,5) BUF(K)(1:NC(K))

C   LIST OTHER INTERNAL PARAMETERS ...
    TBMAXX = TBMX * AMIN1(C LX,CLY)
    UNLAST = UN * AMAX1(C LX,CLY)
    WRITE(IL,20) XO*CLX,YO*CLY,LINES,TBMAXX,UNLAST

```

```

20  FORMAT(/' FIELD ORIGIN relative to the TBM origin =',2G13.6,
1      /' The Number of Turning Band Lines Equals =',I9,
1      /' The Maximum Turning Band Line Length =',G13.6,
1      /' Turning Band Line Discretization Length =',G13.4)

      IF(ICOVF.LE.3 .AND. IULP.NE.2 .AND. ISAJ.EQ.0) NMAX = NHAR
      IF(ICOVF.EQ.4 .OR. IULP.EQ.2) WRITE(IL,30) DS*AMIN1(C LX,CLY)
30  FORMAT(' Discretization Distance for MA Process =',G13.4)
      IF(ICOVF.GE.4 .OR. IULP.EQ.2) WRITE(IL,32) NMAX
32  FORMAT(' Number of Output Points Along each Line =',I9)

      IF(ICOVF.EQ.5 .AND. (A3.NE.0 .OR. A5.NE.0) ) WRITE(IL,35) DT
35  FORMAT(' Discretization Distance for WL Process =',G13.4)

      IF(ICOVF.LE.3 .AND. IULP.LE.1) WRITE(IL,40) FMAX,NHAR,FMAX/NHAR
40  FORMAT(' The Maximum Frequency for the Spectrum =',G13.6,
1      /' Number of Harmonics for the Spectrum =',I9,
1      /' Frequency Spacing in Spectral Domain =',G13.5)

      IF(KS.EQ.1) WRITE(IL,*)
      IF(KS.EQ.2) THEN
        WRITE(IL,45) DX,DY
45  FORMAT(' The Spatial Discretizations, DELX, DELY =',2G13.4)
        SMPLS = (XMAX/CLX)/2.0*(YMAX/CLY)/2.0
        IF(ICOVF.LE.4) WRITE(IL,50) CLX/DX,CLY/DY,SMPLS
50  FORMAT(' No Points/correlation Length in X,Y Dir =',F9.1,
1      3X,F9.1/' Approximate No. of Independent Samples =',F9.1)
      END IF

      RETURN
      END

```

SUBROUTINE MOVAVG(TT,Z1,FF)

```

C-----
C  MOVING AVERAGE SIMULATION OF THE LINE PROCESS (FOR TELIS COVARIANCE)

      REAL    TT(*), Z1(*), FF(*)
      INCLUDE 'tuba211d.inc'

C  FM, FA, NR, UD AND KD ARE ALL CALCULATED IN SUBROUTINE INTPAR.
C  CK IS FOR MOVING AVERAGE PROCESS ASSOCIATED WITH TELIS COV FCN.
C  CK IS SET TO 1.0 IN SUBROUTINE CALINP FOR USER-DEFINED MOVING AVERAGES.

      DO 10 K=1,NR
10     TT(K) = URN55()

      DO 30 N=1,NMAX
        Z1(N) = 0.0
        IOFF = (N-1)*UD + 0.5
        DO 20 K=1,KD
          IADR = IOFF + FLOAT(K)*FM + FA
          Z1(N) = Z1(N) + FF(K)*TT(IADR)
20     CONTINUE
        Z1(N) = CK*Z1(N)
30     CONTINUE

      RETURN
      END

```

FUNCTION NCHR(BUF)

```

C-----
C  DETERMINE THE NUMBER OF CHARACTERS IN THE CHARACTER ARRAY BUF

      CHARACTER BUF*(*)

      LGTH = LEN(BUF)
      DO 10 K=LGTH,1,-1
10     IF(BUF(K:K).NE.' ') GO TO 20
20     NCHR = K

      RETURN
      END

```

```

SUBROUTINE OPNFIL(IT,IL,IO,LREC,ISIM,NSIM,VAR,SSQ,PTS,
1 MODEL,ISEED)
-----
C OPEN DATA OUTPUT FILE AND LIST THE RANDOM FIELD STATISTICS

CHARACTER FMT*35,FNAM*35
CHARACTER FNAME*25,EXT*5,IFM*5,FRM*11
INCLUDE 'tuba211d.inc'
COMMON /OTPTS1/ FMT,FNAM
COMMON /OTPTS2/ IFO,IMO,IRO,ILN
DATA SMBAR,SMSSQ,SMPTS /0.0,0.0,0.0/

NC = NCHR(FNAM)
FNAME(1:NC) = FNAM(1:NC)
FNAME(NC+1:25) = ' '

C APPEND SIMULATION NUMBER TO FILENAME IF MULTIPLE FILES REQUESTED
IF(NF.GT.1) THEN
  IDOT = INDEX(FNAM,'.')
  IF(IDOT.EQ.0) IDOT = NC + 1
  NDIG = ALOG10( FLOAT(NSIM) ) + 1
  WRITE(IFM,10) NDIG
10  FORMAT(2H(I,I1,1H))
  WRITE(EXT,IFM) ISIM
  DO 15 I=1,NDIG-1
15  IF(EXT(I:I).EQ.' ') EXT(I:I) = '0'
  FNAME = FNAM(1:IDOT-1) // '.' // EXT(1:NDIG)
  NC = IDOT + NDIG
  END IF

  IF(NF.GT.1 .OR. ISIM.EQ.1) THEN
    CLOSE(UNIT=IO)
    IF(MOD(IFO,2).EQ.0) FRM = 'UNFORMATTED'
    IF(MOD(IFO,2).NE.0) FRM = 'FORMATTED'
Comt RECL cannot be used for unformatted files with Lahey Fortran
Comt OPEN(UNIT=IO,FILE=FNAME,STATUS='UNKNOWN',FORM=FRM,RECL=LREC)
    OPEN(UNIT=IO,FILE=FNAME,STATUS='UNKNOWN',FORM=FRM)
  END IF

C LIST NEW RANDOM SEED FOR MULTIPLE SIMULATION RUN
IF(NSIM.GT.1) WRITE(IL,18) ISIM,FNAME(1:NC),ISEED
18  FORMAT(/24X,' Simulation Nmbr =',I9,
1     /24X,' Output Filename =',1X,A,
1     /24X,' New Random Seed =',I12)

C LIST THE STATISTICS FOR THE SAMPLE DATA TO LIST FILE AND TERMINAL
WRITE(IL,20) BAR,VAR
20  FORMAT( 24X,' The Sample Mean =',G13.5,
1     /24X,' Sample Variance =',G13.5)
  IF(IPF.NE.1) WRITE(IT,*)
  IF(IPF.NE.1 .AND. NSIM.GT.1) WRITE(IT,28) ISIM
28  FORMAT(' Simulation Nmbr =',I8)
  WRITE(IT,30) FNAME(1:NC),BAR,VAR
30  FORMAT(' Output Filename =',1X,A,
1     /' The Sample Mean =',G13.5,
1     /' Sample Variance =',G13.5/)

  SMBAR = SMBAR + BAR
  SMSSQ = SMSSQ + SSQ
  SMPTS = SMPTS + PTS

C LIST ENSEMBLE STATISTICS IF THIS IS THE LAST REALIZATION
IF(NSIM.GT.1 .AND. ISIM.EQ.NSIM) THEN
  ENBAR = SMBAR/FLOAT(NSIM)
  ENVAR = (SMSSQ-SMPTS*ENBAR*ENBAR)/(SMPTS-1.)
  WRITE(IL,40) ENBAR,ENVAR
40  FORMAT(/22X,' THE ENSEMBLE STATISTICS ...' ,
1     /22X,' The Ensemble Mean =',G13.5,
1     /22X,' Ensemble Variance =',G13.5)
  WRITE(IT,50) ENBAR,ENVAR
50  FORMAT(//' THE ENSEMBLE STATISTICS ...' ,
1     /' The Ensemble Mean =',G13.5,
1     /' Ensemble Variance =',G13.5)
  END IF

RETURN
END

```



```

SUBROUTINE ORGMAX(XY)
C-----
C   CALCULATE DEFAULT TBM ORIGIN AND MAXIMUM DISTANCE ALONG ANY TBM LINE

REAL    XY(2,*)
INCLUDE 'tuba211d.inc'

C   FOR OUTPUT AT ARBITRARY LOCATIONS (KS=1):
C   SET DEFAULT TBM ORIGIN EQUAL TO THE MINIMUM (X,Y) COORDINATE
IF(KS.EQ.1) THEN
    X14 = -1.E-15
    X23 = +1.E+15
    Y12 = -1.E-15
    Y34 = +1.E+15
    DO 10 I=1,NXY
        X14 = AMAX1(XY(1,I),X14)
        X23 = AMIN1(XY(1,I),X23)
        Y12 = AMAX1(XY(2,I),Y12)
        Y34 = AMIN1(XY(2,I),Y34)
10    CONTINUE
    XO = X23
    YO = Y34
    DXX = X14 - XO
    DYY = Y12 - YO
    TBMX = SQRT(DXX*DXX + DYY*DYY)
C-----
C   THE REMAINDER OF THIS IF-BLOCK PERTAINS ONLY TO THE CASE OF GENER-
C   ATING AT ARBITRARY LOCATIONS (KS=1) USING GENERALIZED COVARIANCES.
C   The following is used to calculate DX and DY and ASSUMES a uniform
C   distribution of the "finite element" grid points (i.e., the (x,y)
C   arbitrary locations for generating the field). DX and DY are only
C   needed for this case (KS=1) in subroutine DEFPAR where the DEFAULT
C   TBM line discretization length (UN) is calculated. The calculated
C   value of UN is only APPROXIMATED and should be checked for adequacy
C   (e.g., UN should be .LE. the minimum spacing between any two field
C   generation points). ASSUMPTIONS: (1) NY/NX=DYY/DXX, (2) NX*NY = NXY
C-----
    NX = SQRT( FLOAT(NXY)*DXX/DYY )
    NY = NXY/NX
    DX = 0.2*DXX/FLOAT(NX)
    DY = 0.2*DYY/FLOAT(NY)
END IF

C   FOR GRIDDED OUTPUT (KS=2,3):
C   SET DEFAULT TBM ORIGIN AND FIND THE MAXIMUM DISTANCE FROM
C   THE TBM ORIGIN TO THE FAR CORNER OF THE GRID
IF(KS.GT.1) THEN
    XO = 0.0
    YO = 0.0
    TBMX = SQRT(XMAX*XMAX + YMAX*YMAX)
END IF

RETURN
END

SUBROUTINE OUTPUT(IT,IL,IO,ISIM,NSIM,XY,ZZ,ZM,NEWS)
C-----
C   FINISH FIELD GENERATION, THEN WRITE FIELD TO OUTPUT FILE

CHARACTER  FMT*35,FNAM*35
REAL      ZZ(*), ZM(*), XY(2,*)
INCLUDE 'tuba211d.inc'
COMMON /OTPTS1/ FMT,FNAM
COMMON /OTPTS2/ IFO,IMO,IRO,ILN

C   DO THE FINAL SCALING, ADD IN THE MEAN, AND CALCULATE STATISTICS
CALL FSCALE(XY,ZZ,ZM,ILN,BAR,VAR,SSQ,PTS)

C   OPEN OUTPUT FILE AND LIST RANDOM FIELD SAMPLE STATISTICS
C   LREC = BYTE LENGTH OF UNFORMATTED RECORDS
LREC = 256
IF(MOD(IFO,2).EQ.0) THEN
    LREC = 4*NX
    IF(IMO.EQ.1) LREC = 4*NXY
END IF

```

```

CALL OPNFIL(IT,IL,IO,LREC,ISIM,NSIM,BAR,VAR,SSQ,PTS,ICOVF,NEWS)
C   IFO,IMO AND IRO ARE INTERNAL PARAMETERS CONTROLLING OUTPUT FORMAT;
C   THESE ARE CALCULATED IN SUBROUTINE FILPAR.
C   IFO EVEN,ODD -> UNFORMATTED,FORMATTED RESPECTIVELY
C   IMO = 0 -> WRITE MATRIX OUT LINE BY LINE
C   IMO = 1 -> WRITE MATRIX OUT WITH ONE WRITE STATEMENT

IF(IMO.EQ.1 .OR. KS.EQ.1) THEN
  IF(IFO.EQ.1) WRITE(IO,FMT) (ZZ(K),K=1,NXY)
  IF(IFO.EQ.2) WRITE(IO) (ZZ(K),K=1,NXY)
  IF(IFO.EQ.3) WRITE(IO,FMT) (XY(1,K),XY(2,K),ZZ(K),K=1,NXY)
  IF(IFO.EQ.4) WRITE(IO) (XY(1,K),XY(2,K),ZZ(K),K=1,NXY)
ELSE
  JST =(IRO-1)*NY + (2-IRO)
  JND =(2-IRO)*NY + (IRO-1)
  JNC = 3 - 2*IRO
  DO 20 J=JST,JND,JNC
    IST = (J-1)*NX + 1
    IND = IST + NX - 1
    IF(IFO.EQ.1) WRITE(IO,FMT) (ZZ(K),K=IST,IND)
    IF(IFO.EQ.2) WRITE(IO) (ZZ(K),K=IST,IND)
20  CONTINUE
  END IF

RETURN
END

SUBROUTINE PROGSS(MSG,LU,K,KMAX,INC)
C-----
C   REPORT COMPUTATION PROGRESS

CHARACTER MSG*(*)
COMMON /PGPARS/ PCL,IPG

IF(K.EQ.1) THEN
  PCL = 100./FLOAT(KMAX)
  WRITE(LU,10) MSG,K,INT(PCL)
  IPG = 1
  RETURN
END IF

PCT = 100 * FLOAT(K)/FLOAT(KMAX)
IPC = INT(PCT)
DIF = PCT - PCL

IPR = 0
IF(INC.EQ.0) THEN
  IPR = 1
ELSE IF(KMAX.LE.99) THEN
  IF(DIF.GT.INC) IPR = 1
ELSE
  IF(MOD(IPC,INC).EQ.0 .AND. DIF.GE.2) IPR = 1
END IF
IF(K.EQ.KMAX) IPR = 1

IPG = 0
IF(IPR.EQ.1) THEN
  IPG = 1
  PCT = AMIN1(PCT,99.9)
  PCL = PCT
  IF(PCT.EQ.99.9) PCL = 0.0
  WRITE(LU,10) MSG,K,INT(PCT)
10  FORMAT(A,I8,' ... ( ',I2,' % ) ')
END IF

RETURN
END

```

```

SUBROUTINE PROJCT(L,IT,IU,XY,PP,SS,CC,ZZ,ZM,Z1)
C-----
C   ADD PROJECTIONS FROM THE LTH TBM LINE ONTO OUTPUT FIELD

REAL    XY(2,*), PP(2,*), SS(*), CC(*), ZZ(*), ZM(*), Z1(*)
INTEGER INC(3)
INCLUDE 'tuba211d.inc'
COMMON /PGPARS/ PCL,IPG
CHARACTER PLPD*43,TBLIN*22
DATA    INC /20,5,0/
DATA    TBLIN /' Turning Band Line ...'/
DATA    PLPD /' Projecting Line Process Data ... Point No'/

C   ZERO OUT THE OUTPUT FIELD IF ON TURNING BAND LINE NO 1
IF(L.EQ.1) THEN
  DO 10 K=1,NXY
10    ZZ(K) = 0.0
  END IF

C   FOR OUTPUT AT ARBITRARY (X,Y) LOCATIONS ...
IF(KS.EQ.1) THEN
  CALL PROGSS(TBLIN,IT,L,LINES,INC(IPF))
  PCLSAV = PCL
  IPGSAV = IPG
  DO 20 K=1,NXY
Comt   CALL PROGSS(' Projecting Line Process Data ... Point No',
  IF(IPF*IPGSAV.GE.2) CALL PROGSS(PLPD,IT,K,NXY,INC(IPF-1))
  CALL PROJSB(XY,K,L,K,CC,SS,ZZ,Z1)
20    CONTINUE
  PCL = PCLSAV
  GO TO 50
  END IF

C   FOR OUTPUT ONTO REGULAR OR IRREGULARLY-SPACED GRIDS ...
CALL PROGSS(TBLIN,IT,L,LINES,INC(IPF))
PCLSAV = PCL
IPGSAV = IPG
DO 40 I=1,NY
  READ(UNIT=IU,REC=I) ((PP(K,J),K=1,2),J=1,NX)
  DO 30 J=1,NX
    K = (I-1)*NX + J
Comt   CALL PROGSS(' Projecting Line Process Data ... Point No',
  IF(IPF*IPGSAV.GE.2) CALL PROGSS(PLPD,IT,K,NXY,INC(IPF-1))
  IF(MSK.GT.0 .AND. ZM(K).EQ.0) GO TO 30
  CALL PROJSB(PP,J,L,K,CC,SS,ZZ,Z1)
30    CONTINUE
40    CONTINUE
  PCL = PCLSAV

50    IF(IPF*IPGSAV.GT.1) WRITE(IT,*)
  RETURN
  END

SUBROUTINE PROJSB(A2,J,L,K,CC,SS,ZZ,Z1)
C-----
C   DO THE PROJECTION FOR BOTH GRIDDED AND NON-GRIDDED FIELDS

REAL    A2(2,*), CC(*), SS(*), ZZ(*), Z1(*)
INCLUDE 'tuba211d.inc'

XP = A2(1,J)/CLX + XO
YP = A2(2,J)/CLY + YO
XD = ABS( XP*CC(L) + YP*SS(L) )
N1 = INT(XD/UN)+1
ZZ(K) = ZZ(K) + Z1(N1)

RETURN
END

```

```

SUBROUTINE RDINPT(IN,IT,IL,IU,XY,ZM,MODEL,NLINE,NSIM)
C-----
C CONTROL MODULE FOR READING INPUT PARAMETERS

REAL XY(*),ZM(*)
INCLUDE 'tuba211d.inc'

WRITE(IT,10)
10 FORMAT(//' ++++++++ Program "TUBA (version 2.11d)" +++++++',
1 // ' A Code For Simulating 2D Random Fields',
1 /' Via The Turning Bands Method'/)

C OPEN TEMPORARY FILE (LATER DELETED)
OPEN(UNIT=IL,STATUS='SCRATCH')

C QUERY FOR OUTPUT FIELD PARAMETERS
CALL FLDPAR(IN,IT,IL,IU,XY,XY,ZM)

C QUERY FOR COVARIANCE PARAMETERS
CALL COVPAR(IN,IT,IL,MODEL)

C QUERY FOR TURNING BANDS PARAMETERS
CALL TBMPAR(IN,IT,IL,NLINE,XY)

C MODEL REFERS TO THE LINE PROCESS GENERATION METHOD
C WHEREAS ICOVF REFERS THE THE COVARIANCE MODEL TYPE
C NEXT LINE IS NEEDED IN THE MAIN MODULE (FOR A USER-DEFINED MA PROCESS)
IF(IULP.EQ.2) MODEL = 4

C QUERY FOR OUTPUT FILE PARAMETERS
CALL FILPAR(IN,IT,IL)

C QUERY FOR SIMULATION PARAMETERS
CALL SIMPAR(IN,IT,IL,NSIM)

RETURN
END

```

```

SUBROUTINE RDINTG(IN,IT,IL,IV,NV,ID)
C-----
C READ AND REFLECT INTEGER INPUT DATA

CHARACTER CMT*42, BUF*(*)
INTEGER IV(*)
REAL RV(*)

READ (IN,*) (IV(I),I=1,NV)
WRITE(IT,*) '>>>>>' , (IV(I),I=1,NV)
WRITE(IT,*) ' '

CALL COMENT(ID,CMT,NC)
IF(NV.EQ.1) WRITE(IL,11) IV(1), CMT(1:NC)
IF(NV.EQ.2) WRITE(IL,12) IV(1),IV(2), CMT(1:NC)
11 FORMAT(2X, I12,T36,A)
12 FORMAT(2X,2I12,T36,A)

RETURN

```

```

ENTRY RDREAL(IN,IT,IL,RV,NV,ID)
C-----
C READ AND REFLECT REAL INPUT DATA

READ (IN,*) (RV(I),I=1,NV)
WRITE(IT,*) '>>>>>' , (RV(I),I=1,NV)
WRITE(IT,*) ' '

CALL COMENT(ID,CMT,NC)
IF(NV.EQ.1) WRITE(IL,21) RV(1), CMT(1:NC)
IF(NV.EQ.2) WRITE(IL,22) RV(1),RV(2), CMT(1:NC)
IF(NV.EQ.3) WRITE(IL,23) RV(1),RV(2),RV(3),CMT(1:NC)
21 FORMAT(2X, G13.5,T36,A)
22 FORMAT(2X,2G13.5,T36,A)
23 FORMAT(1PE11.3,2E11.3,T36,A)

RETURN

```

```

ENTRY RDCHAR(IN, IT, IL, BUF, ID)
C-----
C   READ AND REFLECT CHARACTER VARIABLES

READ(IN,30) BUF
30  FORMAT(A)
    NB = NCHR(BUF)
    IF(NB.EQ.0) BUF = ' '
    IF(NB.EQ.0) NB = 1
    WRITE(IT,*) '>>>>' , BUF(1:NB)
    WRITE(IT,*) ' '

    CALL COMENT(ID,CMT,NC)
    WRITE(IL,35) BUF(1:NB),CMT(1:NC)
35  FORMAT(A,T36,A)

    RETURN
    END

SUBROUTINE SIMPAR(IN, IT, IL, NSIM)
C-----
C   READ SIMULATION PARAMETERS

SAVE      ISEED, JSEED
CHARACTER BUF*32
INCLUDE 'tuba211d.inc'

WRITE(IT,10)
10  FORMAT(//' ++++++ SIMULATION PARAMETERS ++++++')

WRITE(IT,*) '(1) - Marsaglia and Bray Random Number Generator'
WRITE(IT,*) '(2) - Machine Independent Random Number Generator'
WRITE(IT,*)
CALL RDINTG(IN, IT, IL, IURN, 1, 28)

WRITE(IT,*) 'Enter Integer Seed(s) To Initialize The Generator'
IF(IURN.EQ.2) WRITE(IT,20)
20  FORMAT(' Seed For This Generator Must Be 8 Digits Long ')
WRITE(IT,*)
Comt CALL RDINTG(IN, IT, IL, ISEED, 1, 29)
CALL RDCHAR(IN, IT, IL, BUF, 29)
READ(BUF,*,END=21) ISEED, JSEED
GO TO 30
21  JSEED = ISEED
30  IF(IURN.EQ.1) DUMY = UNITMB(ISEED)
    IF(IURN.EQ.2) DUMY = UNITSS(IT, ISEED)

WRITE(IT,*) 'Enter The Number Of Realizations To Be Simulated'
WRITE(IT,*)
CALL RDINTG(IN, IT, IL, NSIM, 1, 30)

IF(NSIM.GT.1) THEN
    WRITE(IT,*) '(1) - All Realizations Written To One File'
    WRITE(IT,*) '(2) - A Separate File For Each Realization'
    WRITE(IT,*) '(e.g., file.1, file.2 ... file.99)'
    WRITE(IT,*)
    CALL RDINTG(IN, IT, IL, NF, 1, 34)
END IF

WRITE(IT,*) '(0) - Do NOT Artificially Scale The Realizations'
WRITE(IT,*) '(1) - Scale Data To Match Mean and Variance Exactly'
WRITE(IT,*) 'Please Read Section 2.1.2 Of The Manual'
WRITE(IT,*) 'Before Choosing This Option'
WRITE(IT,*)
CALL RDINTG(IN, IT, IL, IMSEX, 1, 36)

IF(IMSEX.EQ.1 .AND. ICOVF.EQ.5) THEN
    WRITE(*,*) 'GC Model: Enter Desired Mean, Nugget and Sill'
    WRITE(IT,*)
    CALL RDREAL(IN, IT, IL, AM, 3, 37)
END IF

WRITE(IT,*) 'Specify The Level Of Status Reporting To The Screen'
WRITE(IT,*) '(1) - Minimal (e.g., For Many Realizations)'

```

```

WRITE(IT,*)'(2) - More Frequent (e.g., For Many TBM Lines)'
WRITE(IT,*)'(3) - Very Frequent (e.g., For Very Large Fields)'
WRITE(IT,*)
CALL RDINTG(IN,IT,IL,IPF,1,35)

RETURN

ENTRY RESEED(ISIM,NUSEED)
C-----
C RESEED THE RANDOM NUMBER GENERATOR

IF(ISIM.EQ.1) THEN
  NUSEED = ISEED
  IF(JSEED.EQ.ISEED) RETURN
  IF(IURN.EQ.1) DUMY = UNITMB(JSEED)
  IF(IURN.EQ.2) DUMY = UNITSS(IT,JSEED)
  RETURN
END IF

NUSEED = 1.E+08*(URN55()+0.5)
IF(IURN.EQ.1) DUMY = UNITMB(NUSEED)
IF(IURN.EQ.2) THEN
36 IF(ALOG10(FLOAT(NUSEED)) .LT. 7) THEN
  NUSEED = 10*NUSEED
  GO TO 36
END IF
  DUMY = UNITSS(IT,NUSEED)
END IF

RETURN
END

SUBROUTINE SPCTRL(L,L2,PA,Z1,DZ,S1,C1,S2,C2)
C-----
C SPECTRAL SIMULATION OF THE LINE PROCESSES

PARAMETER (TUPI=6.283185308)
COMPLEX DZ(*)
REAL PA(*),Z1(*),S1(*),C1(*),S2(*),C2(*)
INCLUDE 'tuba211d.inc'

C GENERATE LINE PROCESS USING THE FFT METHOD
IF(ISAJ.EQ.0) CALL FFTGEN(L,L2,PA,Z1,DZ)
IF(ISAJ.EQ.0) RETURN

C GENERATE LINE PROCESS USING THE METHOD OF SHINOZUKA AND JAN
DO 10 M=1,NHAR
  THETA = URN55() * TUPI
  C2(M) = COS(THETA)
  S2(M) = SIN(THETA)
10 CONTINUE
IF(IPAA.EQ.2) READ(UNIT=L2,REC=L) (PA(M),M=1,NHAR)

C PA(M) = 2.0 * SQRT(SPECTRL DENSITY * DELTA OMEGA)
C C2SAV ETC IS FOR TRIG IDENTITIES - THE COS(OMEGA'*ZETA+PHI) TERM IS
C CALCULATED BY CONSIDERING ZETA = N*DELTA-ZETA AND TRIG IDENTITIES
DO 30 N=1,NMAX
  Z1(N) = 0.0
  DO 20 M=1,NHAR
    Z1(N) = Z1(N)+ PA(M) * C2(M)
    C2SAV = C2(M)
    S2SAV = S2(M)
    C2(M) = C2SAV*C1(M) - S2SAV*S1(M)
    S2(M) = S2SAV*C1(M) + C2SAV*S1(M)
20 CONTINUE
30 CONTINUE

RETURN
END

```

FUNCTION SPDF(FRQ,ICOVF)

```

C-----
C  CALCULATE NORMALIZED 1D SPECTRAL DENSITY FUNCTION FOR POINT PROCESSES
C  HAVING 2D COVARIANCE FUNCTIONS OF: USER-SPECIFIED (ICOVF=0),
C  EXPONENTIAL (ICOVF=1), GAUSSIAN (ICOVF=2), BESSEL (ICOVF=3)

DATA  SOME,THING /0.,1./

IF(ICOVF.EQ.0) THEN
C  USER DEFINED SPECTRUM GOES HERE
  SPDF = SOME + THING
  RETURN
END IF

C  EXPLANATION OF ANCIENT FORTRAN: (COMPUTED GO TO)
C  IF(ICOVF.EQ. 1,2,3) THEN GO TO  (10,20,30)

GO TO (10,20,30) ICOVF

10  DENOM =(1.+FRQ*FRQ)**1.5
    SPDF  = 0.5*FRQ/DENOM
    RETURN

20  XARG = 0.25*FRQ*FRQ
    SPDF = 0.25*FRQ*EXP(-XARG)
    RETURN

30  DENOM = (1.+FRQ*FRQ)**2.0
    SPDF  = 1.0*FRQ/DENOM
    RETURN

END

```

SUBROUTINE TBMPAR(IN,IT,IL,NLINE,XY)

```

C-----
C  QUERY FOR TURNING BANDS LINE PARAMETERS

PARAMETER (PI=3.141592654)
REAL      XY(*)
INCLUDE  'tuba211d.inc'

WRITE(IT,10)
10  FORMAT('//' ++++++++ TURNING BANDS PARAMETERS ++++++')

WRITE(IT,*)'Enter The Number Of Turning Band Lines'
WRITE(IT,*)'      Use At Least 16  '
WRITE(IT,*)
CALL RDINTG(IN,IT,IL,LINES,1,15)
NLINE = LINES

IF(ICOVF.EQ.0) THEN
  WRITE(IT,*)'(1) - Line Process By A Spectral Method'
  WRITE(IT,*)'(2) - Line Process By A Moving Average Method'
  WRITE(IT,*)
  CALL RDINTG(IN,IT,IL,IULP,1,13)
ELSE
  WRITE(IT,*)'For The Remaining Turning Band Parameters:'
  WRITE(IT,*)'(1) - Use Default Turning Band Parameters'
  WRITE(IT,*)'(2) - Enter The TBM Parameters Manually'
  WRITE(IT,*)
  CALL RDINTG(IN,IT,IL,IDFP,1,14)
  IF(IDFP.EQ.1) CALL DEFPAR(NLINE,XY)
  IF(IDFP.EQ.1) RETURN
END IF

WRITE(IT,*)'Enter The TBM Line Discretization Distance'
WRITE(IT,*)'  e.g., Smaller Than The Grid Spacing'
IF(ICOVF.NE.5) THEN
  WRITE(IT,*)'e.g., 1/16th the Correlation Length'
END IF
WRITE(IT,*)
CALL RDREAL(IN,IT,IL,UN,1,16)

IF(ICOVF.LE.3 .AND. IULP.NE.2) THEN
  WRITE(IT,*)'Enter NBR Of Harmonics For Discretizing Spectrum'

```

```

WRITE(IT,*)
CALL RDINTG(IN,IT,IL,NHAR,1,17)
C SAJ METHOD BY DEFAULT, FFT METHOD IF NHAR=2**N FOR SOME N
ISAJ = 1
HMCS = NHAR
16 IF (HMCS/2 .GT. 1) THEN
    HMCS = HMCS/2
    GO TO 16
END IF
IF (HMCS.EQ.2) THEN
    FMAX = 2.*PI/(UN/AMAX1(C LX,CLY))
    ISAJ = 0
END IF
IF (ISAJ.EQ.1) THEN
    WRITE(IT,*)'Enter Max Frequency For Truncation Of Spectrum'
    WRITE(IT,*)
    CALL RDREAL(IN,IT,IL,FMAX,1,18)
END IF
END IF

IF (ICOVF.EQ.4 .OR. IULP.EQ.2) THEN
    WRITE(IT,*)'Enter Discretization Distance for the MA Process'
    WRITE(IT,*)' Suggest 1/20th Of The Correlation Length'
    WRITE(IT,*)'and No Larger Than 1/10th TBM Line Disc. Dist.'
    WRITE(IT,*)
    CALL RDREAL(IN,IT,IL,DS,1,19)
END IF

DT = UN
IF (ICOVF.EQ.5 .AND. (A3.NE.0 .OR. A5.NE.0) ) THEN
    WRITE(IT,*)'Enter Discretization Distance for Weiner Process'
    WRITE(IT,*)'Suggest 1/5th Of The TBM Discretized Distance'
    WRITE(IT,*)
    CALL RDREAL(IN,IT,IL,DT,1,20)
END IF

WRITE(IT,*)'Enter (Xo,Yo) Field Origin Relative To TBM Origin'
WRITE(IT,*)
CALL RDREAL(IN,IT,IL,XO,2,21)

WRITE(IT,*)'Enter the Maximum Turning Band Line Length'
WRITE(IT,*)
CALL RDREAL(IN,IT,IL,TBMX,1,31)

RETURN
END

```

```

FUNCTION UNITMB(ISEED)

```

```

C-----
C URN01 generates UNIFORM RANDOM NUMBERS on the interval [0,1] using the
C algorithm of Marsaglia and Bray presented in (pages 567 & 597) of:
C
C "The Handbook of Random Number Generation and Testing
C with TESTRAND computer code"
C E. J. Dudewicz and T. G. Rally
C American Sciences Press, Inc. 1981.
C
C This generator was "recommended for practical use" (page 134) by the
C above authors. This generator passed the very sensitive and exhaustive
C tests described in the above reference.
C-----
C NOTE !!! Compile with "integer overflow check" turned OFF
C-----
C This version of Marsaglia's code was arranged by D. A. Zimmerman
C at New Mexico Tech, Geoscience Dept., Hydrology Program, August, 1987.
C-----
C DUMY = URNIT(ISEED) ! Initialize Random Number Generator
C DO 10 I=1,N ! Generate N Uniformly Distributed
C 10 X(I) = URN01() ! Random Numbers On Interval [0,1]
C-----
INTEGER N1(64), N2(64), N(128), MS(6)
EQUIVALENCE (N(1),N1), (N(65),N2), (MS(1),ML)
COMMON /SEEDS/ ML,MM,MK,L,M,K
C*** SEE BLOCK DATA MODULE

```



```
C*** DATA L, M, K /089347405, 301467177, 240420681/
C*** DATA ML,MM,MK /65539, 33554433, 36243609/
```

```
DATA N1/ 880431333, 845941495, 233211304, 1989552121,
1 465185814, 280672924, 294923811, 969688974, 798989604,
1 379880543, 130022074, 1958997525, 1074191695, 680854387,
1 751282651, 1208899767, 695831691, 1667008051, 1682546364,
1 1984522335, 287570376, 1137852001, 1597983496, 2015817872,
1 1479672206, 1468443024, 1657203843, 326324124, 680973716,
1 1451006002, 1251441372, 241092947, 1815086916, 1807193097,
1 770906592, 725422944, 1822111098, 470585328, 939566271,
1 1084841038, 1988336409, 229735215, 1763201387, 2072973152,
1 1143606610, 548108569, 544252510, 1980873641, 1195919839,
1 2089487851, 1406149582, 1839198022, 2106705200, 189238196,
1 1170370207, 1304402631, 1936129483, 810953177, 706509560,
1 476957499, 1307077413, 824336639, 1487297852, 1591453718/
```

```
DATA N2/ 1348888685, 155452792, 265840413, 1440038626,
1 770186799, 1152058296, 1726999383, 1389732859, 1838014251,
1 1751063044, 102451305, 212848938, 1046489181, 976388856,
1 1797117421, 461971124, 259337424, 492056652, 1152625277,
1 1087711027, 344810019, 1477716555, 809152324, 1766452264,
1 1687482934, 1077592551, 1906112218, 328744821, 1380339247,
1 339750038, 1993648985, 1054008271, 2006727977, 1618648061,
1 1300903972, 168650429, 1734500183, 906733794, 614096451,
1 1092917209, 1180334545, 577024776, 1406305431, 648073629,
1 973807028, 883884075, 1562357277, 1705648154, 1377603620,
1 1845151798, 220566094, 768813055, 571717967, 218994012,
1 212872559, 1824677815, 1573937649, 450149130, 284847256,
1 2062965934, 47834840, 1766553923, 1580332201, 182920702/
```

```
I = MOD(ISEED,6) + 1
MS(I) = ISEED
RETURN
```

```
C ENTRY URN01()
ENTRY URNMB()
```

```
C-----
C URN01 RETURNS UNIFORMLY DISTRIBUTED RANDOM NUMBERS ON [0.,1.]
```

```
L = ML * L
M = MM * M
K = MK * K
J = 1.0 + IABS(L) / 16777216
```

```
C URN01 = 0.5 + FLOAT( N(J)+L+M ) * 0.23283064E-09
URNMB = FLOAT( N(J)+L+M ) * 0.23283064E-09
N(J) = K
```

```
RETURN
END
```

```
FUNCTION UNITSS(LU,ISEED)
```

```
C-----
C URN01 generates UNIFORM RANDOM NUMBERS on the interval [0,1]
```

```
C
C Reference: C. G. Swain and M. S. Swain 1980. "A Uniform
C Random Number Generator That Is Reproducible,
C HARDWARE-INDEPENDENT, And Fast" J. Chem. Inf.
C Comput. Sci. Vol 20. pp 56-58.
```

```
C
C According to E. J. Dudewicz, Dept. Statistics, Ohio State University,
C in "Modern and Easy Generation of Random Numbers / Testing of Random
C Number Generators with TESTRAND", 10th IMACS World Congress On System
C Simulation and Scientific Computation, August 8-13, 1982, Montreal,
C Canada, Proceedings, Volume 2, page 133, this generator failed the
C sensitive Chi-square on Chi-square test performed by the TESTRAND code.
```

```
C-----
C This version of Swain & Swain's code was arranged by D. A. Zimmerman
C at New Mexico Tech, Geoscience Dept., Hydrology Program, June, 1986.
```

```
C-----
C DUMY = URNIT(12345678) ! INITIALIZE URNG: ISEED=12345678
C DO 10 K=1,N ! GENERATE N UNIFORMLY DISTRIBUTED
C 10 X(K) = URN01() ! RANDOM NUMBERS ON INTERVAL [0,1]
```

```

PARAMETER (K1=35260417, K2=72619094, K3=86952743)
INTEGER    M(0:3)
DATA       M /0,K1,K2,K3 /

SEED = FLOAT(ISEED)
IPWR = ALOG10(SEED)

IF(IPWR.NE.7) THEN
  WRITE(LU,*)' ***** URN GENERATOR SEED MUST BE 8 DIGITS LONG'
  WRITE(LU,*)'          PROGRAM EXECUTION HALTED'
  STOP
END IF

M(1) = K1
M(2) = K2
M(3) = K3

I = MOD(ISEED,3) + 1
M(I) = ISEED

RETURN

C  ENTRY URN01()
C  ENTRY URNSS()
-----
C  URN01 RETURNS UNIFORMLY DISTRIBUTED RANDOM NUMBERS ON [0.,1.]

M(0) = M(1) + M(2) + M(3)

IF(M(2) .LT. 50000000) M(0) = M(0) + 1357
IF(M(0) .GE. 100000000) M(0) = M(0) - 100000000
IF(M(0) .GE. 100000000) M(0) = M(0) - 100000000

M(1) = M(2)
M(2) = M(3)
M(3) = M(0)

C  URN01 = 1.0E-08 * M(0)
C  URNSS = 1.0E-08 * M(0) - 0.5

RETURN
END

FUNCTION URNAB(A,B)
-----
C  URNAB returns uniformly distributed random numbers on [A,B]

comt URNAB = A + (B-A) * URN01()
      URNAB = A + (B-A) * ( URN55() + 0.5 )

RETURN
END

FUNCTION URN55()
-----
C  RETURN UNIFORMLY DISTRIBUTED RANDOM NUMBER ON INTERVAL [-.5,+ .5]

INCLUDE 'tuba211d.inc'

C  IURN = 1  MARSAGLIA AND BRAY RANDOM NUMBER GENERATOR (RECOMMENDED)
C  IURN = 2  SWAIN AND SWAIN MACHINE INDEPENDENT RANDOM NUMBER GENERATOR

IF(IURN.EQ.1) URN55 = URNMB()
IF(IURN.EQ.2) URN55 = URNSS()

RETURN
END

```

```

SUBROUTINE WNRLVY(Z1)
C-----
C   NON STATIONARY CASE: WIENER-LEVY SIMULATION OF LINE PROCESS

REAL    Z1(*)
INCLUDE 'tuba211d.inc'

Z1(1) = 0.0
W1 = 0.0
AI1 = 0.0
BI1 = 0.0
TT = 0.0
DO 20 N=2,NMAX
  DO 10 K=1,KT
    TT = TT+DT
    W2 = W1 + SG * URN55()
    AI1 = AI1 + 0.5*(W2+W1)*DT
    BI1 = BI1 + 0.5*(W1*(TT-DT)+TT*W2)*DT
    W1 = W2
10  CONTINUE
    Z1(N) = B0*W1 + (B1+B2*TT)*AI1 - B2*BI1
20  CONTINUE

RETURN
END

```

```

FUNCTION WTEXP(OM,ACL,ASL,AL1,AL2)
C-----
C   CALCULATE SPECTRAL DENSITY WEIGHTS FOR EXPONENTIAL AREAL AVERAGE PROCESS

AS = SIN(AL1*OM*ACL/2.)
BS = SIN(AL2*OM*ASL/2.)

IF(ACL.NE.0. .AND. ASL.NE.0.) THEN
  AL12 = AL1*AL1*AL2*AL2
  OMOM = OM*OM*OM*OM
  A1B1 = ACL*ACL*ASL*ASL
  ASBS = AS*AS*BS*BS
  ALOA = AL12*OMOM*A1B1
  WTEXP = ASBS*(16./ALOA)
END IF

IF(ACL.EQ.0.) WTEXP = BS*BS*(4./(AL2*AL2*ASL*ASL*OM*OM))
IF(ASL.EQ.0.) WTEXP = AS*AS*(4./(AL1*AL1*ACL*ACL*OM*OM))

RETURN
END

```

```

FUNCTION WTUSR(OM,ACL,ASL,AL1,AL2)
C-----
C   RETURN SPECTRAL DENSITY WEIGHTS FOR USER-DEFINED AREAL AVERAGE PROCESS

DATA    SOME,THING /0.,1./

C   SEE CHAPTERS 2 AND 5 OF THE DOCUMENTATION
WTUSR = SOME + THING

RETURN
END

```

