Ph.D. Thesis

Computations With Gaussian Random Fields

by Boris Kozintsev
Advisor: Benjamin Kedem

Ph.D. 99-3

ABSTRACT

Title of Dissertation: COMPUTATIONS WITH GAUSSIAN RANDOM FIELDS

Boris Kozintsev, Doctor of Philosophy, 1999

Dissertation directed by: Professor Benjamin Kedem
Department of Mathematics

An approach to computational problems associated with generation and estimation of large Gaussian fields is studied. Fast algorithms for matrix operations on circulant matrices are presented, and a connection between such matrices and covariance matrices of Gaussian fields is established.

Based on this approach, a model for discrete spatial data is introduced, extending the work of Nott and Wilson (1997). We assume that discrete random fields are obtained by clipping a stationary zero mean Gaussian random field at several fixed levels. The model is defined by this set of levels, a choice of a family of covariance functions for the Gaussian field, and a parameter vector specifying a particular covariance function within the family.

For this model, the Stochastic Expectation-Maximization algorithm for estimating the covariance parameter vector is presented. The algorithm includes conditional generation of Gaussian fields given that components fall within specified intervals; this is achieved by the Gibbs sampler - a Markov Chain Monte Carlo technique.

The precision of the algorithm – understood in terms of the variance of the resulting estimator of the correlation function – is compared to that of estimating the parameter directly from the Gaussian data by the Maximum Likelihood method. For this purpose, the Fisher information matrix in the Gaussian model is computed, the asymptotic distribution of the MLE estimator of the correlation parameter is established, and simulations are performed to compare the empirical variances of the MLE and several SEM estimators (the latter based on various quantizations) to the variance predicted by the theory, and to each other.

COMPUTATIONS WITH GAUSSIAN RANDOM FIELDS

by

Boris Kozintsev

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland at College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
1999

DEDICATION

To my parents and Sandra

ACKNOWLEDGEMENTS

I wish to express my deepest gratitude to Professor Benjamin Kedem, whom I was lucky to have as an advisor. His enthusiasm towards this research, together with his friendship, made the whole thing possible.

I want to thank Dr. von Petersdorff for granting me access to the fast DEC Alpha computers for simulations, and for his Scientific Computations course (MAPL 660).

I also want to thank Victor de Oliveira and David Bindel, whose BTG project was a source of inspiration and useful subroutines.

I appreciate the financial support from the NASA Grant NAG52783.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Gaussian Random Fields and Computations

In the study and analysis of spatial data from sciences such as geology, geography, hydrology, and meteorology, to name a few, a quantity of interest, say z, varies over a domain D in space according to an unknown function $z : D \subset R^d \to R$. Usually, z is observed only in a small number of locations in D, and inference about z is then based on a proposed mathematical model for the function $z(.)$.

The stochastic approach to modeling $z(.)$ is to view it as a realization of a random field. Many examples of successful application of this approach in a variety of situations are contained in Cressie [15] and Hjort and Omre [30].

Perhaps the most important role in the stochastic approach is played by Gaussian random fields. Many natural phenomena are modeled directly as realizations of Gaussian fields; in other cases, not the data themselves, but some nonlinear transformation thereof is assumed normal (for example, de Oliveira et al., [48]). Even discrete data can be modeled, as shown in Chapter 4 of this thesis, as levels of some unobserved Gaussian field.

The usefulness of the Gaussian model comes from two directions. First, many datasets from the natural sciences display markedly Gaussian characteristics, probably because of the Central Limit Theorem, or for some other reasons. Secondly, Gaussian random fields are well known and convenient mathematical objects, defined completely by their mean and covariance functions.

The Gaussian model is especially practical when we are interested only in a few locations in D, for example, in cases when values of the field at several locations are available as observations, and it is desirable to predict (interpolate) the field at several other locations. Then both observed and unobserved values can be pooled together in a multivariate Gaussian vector of a reasonable length, whose distribution immediately becomes known, and further analysis can be carried out on this vector only, regardless of the size of the original field.

This approach brings no simplification, however, when we are interested in the values of the field 'everywhere,' that is, on some fine grid in D – for example, in texture generation and analysis related applications. In such cases, the multivariate Gaussian vector becomes 'too long' to work with, because storage and computational requirements for dealing with its covariance matrix become prohibitive. In particular, it is impossible just to generate the Gaussian field on a large grid using the standard (and, until recently, the only) method for generating multivariate normal vectors based on Cholesky decomposition of the covariance matrix.

The last fact is, of course, unfortunate, since being able to generate Gaussian fields routinely is clearly very important for the search for and evaluation of models and algorithms.

However, several years ago the possibility of using the so-called circulant embedding technique (known in the field of computational linear algebra) in connection with stationary Gaussian fields was pointed out (Dembo et al. [18], Davies and Harte [17], Dietrich and Newsam [22], Wood and Chan [63]).

The primary motivation of this thesis is to use this new technique for building and evaluating a model for discrete spatial data.

1.2 Thesis Synopsis

We now give a synopsis of the thesis chapter by chapter.

Chapter 2 discusses computational linear algebra and lays the foundation for the rest of the thesis. We introduce two important classes of matrices – block Toeplitz and block circulant. It turns out that while the former occur naturally in statistics as covariance matrices of samples from stationary processes over regular grids, the latter have a unique computational property of being diagonalizable by the Fourier transform. This simplifies greatly almost any imaginable matrix operation, such as multiplication, taking the inverse, computing the determinant, and evaluating quadratic forms. Moreover, storage requirements when working with $n \times n$ circulant matrices are of the order n, rather than n^2. Toeplitz and circulant matrices are related in the sense that a Toeplitz matrix can always be embedded into a larger circulant matrix in a standard way.

Chapter 3 discusses random processes and fields and explains how circulant embedding from Chapter 2 can be used for fast and exact generation of large samples from the Gaussian fields. Also, a method is presented for conditional generation of such samples given that certain (or all) components fall within specified intervals. This is necessary for working with the discrete spatial data model described in Chapter 4.

Chapter 4 describes a modeling device, in which discrete spatial data are treated as a quantized unobserved Gaussian field. Using the technique from Chapters 2 and 3, the computationally tractable SEM algorithm for estimation of the model parameters is presented. Examples of the modeled data and estimation results are provided.

In Chapter 5 we study how much information would be available to us in the model introduced in Chapter 4, if we had the 'original,' non-quantized Gaussian data. For this purpose we compute the Fisher information matrix and hence the lower bound on the variance of the estimator. We also find the asymptotic distribution of the estimator when the grid size becomes large (but we still have only one realization of the field available as data). We compare this theoretical variance to the actual variance of the MLE estimator from the Gaussian data as observed in simulations, and also to the variance of the SEM estimators obtained from the same data clipped at various numbers of thresholds.

Chapter 6 summarizes the main contributions of this thesis and formulates possible directions for future research.

Chapter 2

COMPUTATIONAL PROPERTIES OF SYMMETRIC BLOCK CIRCULANT MATRICES WITH CIRCULANT BLOCKS

2.1 Introduction

In this chapter we describe a class of matrices that resemble sparse matrices in the sense that the matrix-vector multiplication can be computed efficiently, and that the required storage for them is significantly less than n^2. We also describe algorithms for inverting these matrices, and computing determinants and quadratic forms. These algorithms will be applied in the next chapters to the covariance matrices of samples from Gaussian random fields, for generating such samples and for evaluating their likelihoods.

2.2 Toeplitz and Circulant matrices

Definition 1[1]. A matrix $\mathbf{T}$ is called Toeplitz if it has constant values along diagonals:

$$\mathbf{T} = \begin{pmatrix} t_0 & t_1 & t_2 & \dots & t_{n-1} \\ t_{-1} & t_0 & t_1 & \ddots & \vdots \\ t_{-2} & t_{-1} & t_0 & \ddots & t_2 \\ \vdots & \ddots & \ddots & \ddots & t_1 \\ t_{-(n-1)} & \dots & t_{-2} & t_{-1} & t_0 \end{pmatrix}$$

Each $n \times n$ Toeplitz matrix has at most $2n-1$ different entries. It is defined completely by its first column and first row.

Definition 2. A matrix $\mathbf{C}$ is called circulant if its columns are circular shifts of the first column:

$$\mathbf{C} = \begin{pmatrix} c_0 & c_{n-1} & c_{n-2} & \dots & c_1 \\ c_1 & c_0 & c_{n-1} & \ddots & \vdots \\ c_2 & c_1 & c_0 & \ddots & c_{n-2} \\ \vdots & \ddots & \ddots & \ddots & c_{n-1} \\ c_{n-1} & c_{n-2} & c_{n-3} & \dots & c_0 \end{pmatrix}$$

A circulant $n \times n$ matrix has at most n different entries and is determined completely by its first column (or row): all the consecutive columns (or rows) are obtained by shifting the first one circularly, that is, in such a way that the last element becomes first and all the other elements are shifted forward by one.

All circulant matrices are also Toeplitz, but not vice versa.

Block Toeplitz and block circulant matrices are defined similarly, but the structure applies to blocks rather than to individual entries. In general there is

[1]In this section we follow Sjöström [56].

no restriction on the structure of the blocks. However, for the purposes of this dissertation, we are only interested in block Toeplitz (block circulant) matrices that also have Toeplitz (circulant) structure within each block.

Symmetric positive definite block Toeplitz matrices with Toeplitz blocks are important for us because, with the appropriate numbering of the grid sites, such is the structure of covariance matrices of stationary random fields observed over regular grids. In particular, an $n_1 \times n_2$ grid yields an $n_1 n_2 \times n_1 n_2$ covariance matrix with n_1 blocks, each of size $n_2 \times n_2$.

However, efficient computational methods require a more specialized block circulant structure. Therefore we embed our block Toeplitz matrices into larger block circulant ones, as described in the next section.

2.3 Circular embedding

Suppose we want to embed a symmetric block Toeplitz matrix $\mathbf{T}$. Here is its general form:

$$\mathbf{T} = \begin{pmatrix} \mathbf{T}^{(1)} & \mathbf{T}^{(2)} & \dots & \mathbf{T}^{(n_1)} \\ \mathbf{T}^{(2)^T} & \mathbf{T}^{(1)} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \mathbf{T}^{(n_1)^T} & \dots & \dots & \mathbf{T}^{(1)} \end{pmatrix}$$

The n_1 different Toeplitz blocks $\mathbf{T}^{(i)}$ of size $n_2 \times n_2$ each look like this:

$$\mathbf{T}^{(i)} = \begin{pmatrix} t_0^{(i)} & t_1^{(i)} & t_2^{(i)} & \cdots & t_{n_2-1}^{(i)} \\ t_{-1}^{(i)} & \ddots & \ddots & \ddots & \vdots \\ t_{-2}^{(i)} & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ t_{-(n_2-1)}^{(i)} & \cdots & \cdots & \cdots & \ddots \end{pmatrix}, \quad i = 1, \ldots, n_1$$

In addition, $\mathbf{T}^{(1)}$ has to be symmetric.

We start embedding $\mathbf{T}$ into $\mathbf{C}$ by embedding each $\mathbf{T}^{(i)}$ into a circulant $2n_2 \times 2n_2$ matrix $\mathbf{C}^{(i)}$. The first column of $\mathbf{C}^{(i)}$ has the same entries as the first column of $\mathbf{T}^{(i)}$, followed by a zero and the inverted first row of $\mathbf{T}^{(i)}$ less the first element.

$$\mathbf{C}^{(i)} = \begin{pmatrix} t_0^{(i)} & \cdots & \cdots & \cdots \\ t_{-1}^{(i)} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ t_{-(n_2-1)}^{(i)} & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & \cdots \\ t_{n_2-1}^{(i)} & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ t_2^{(i)} & \cdots & \cdots & \cdots \\ t_1^{(i)} & \cdots & \cdots & \cdots \end{pmatrix} = \begin{pmatrix} \mathbf{T}^{(i)} & ... \\ ... & \mathbf{T}^{(i)} \end{pmatrix}$$

This defines the circulant $\mathbf{C}^{(i)}$ completely, since all the other columns are the circulations of the first one. Next we combine the $\mathbf{C}^{(i)}$'s into a block circulant $4n_1n_2 \times 4n_1n_2$ matrix $\mathbf{C}$. As before, we specify the first block column of $\mathbf{C}$ only, and the rest of $\mathbf{C}$ is obtained by circulation:

$$\mathbf{C} = \begin{pmatrix} \mathbf{C}^{(1)^T} & \dots & \dots & \dots & \dots \\ \mathbf{C}^{(2)^T} & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ \mathbf{C}^{(n_1)^T} & \dots & \dots & \dots & \dots \\ \mathbf{0}_{2n_2 \times 2n_2} & \dots & \dots & \dots & \dots \\ \mathbf{C}^{(n_1)} & \dots & \dots & \dots & \dots \\ \mathbf{C}^{(n_1-1)} & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ \mathbf{C}^{(2)} & \dots & \dots & \dots & \dots \end{pmatrix}$$

where $\mathbf{0}$ is a matrix of zeros. By definition, $\mathbf{C}$ is block circulant with circulant blocks, and also symmetric.

In short, to embed a symmetric block Toeplitz matrix $\mathbf{T}$, we embed each of its blocks $\mathbf{T}^{(i)}$ into a circulant $\mathbf{C}^{(i)}$, and then combine the resulting $\mathbf{C}^{(i)}$'s into symmetric block circulant $\mathbf{C}$.

This is how $\mathbf{C}$ looks like in terms of the original $\mathbf{T}^{(i)}$'s:

$$\mathbf{C} = \begin{pmatrix} \mathbf{T}^{(1)} & \dots & \mathbf{T}^{(2)} & \dots & \mathbf{T}^{(3)} & \dots & \dots & \dots \\ \dots & \mathbf{T}^{(1)} & \dots & \mathbf{T}^{(2)} & \dots & \mathbf{T}^{(3)} & \dots & \dots \\ \\ \mathbf{T}^{(2)^T} & \dots & \mathbf{T}^{(1)} & \dots & \mathbf{T}^{(2)} & \dots & \dots & \dots \\ \dots & \mathbf{T}^{(2)^T} & \dots & \mathbf{T}^{(1)} & \dots & \mathbf{T}^{(2)} & \dots & \dots \\ \\ \mathbf{T}^{(3)^T} & \dots & \mathbf{T}^{(2)^T} & \dots & \mathbf{T}^{(1)} & \dots & \dots & \dots \\ \dots & \mathbf{T}^{(3)^T} & \dots & \mathbf{T}^{(2)^T} & \dots & \mathbf{T}^{(1)} & \dots & \dots \\ \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix}$$

We now illustrate circulant embedding on example with $n_1 = 2$ and $n_2 = 3$. Consider a symmetric block Toeplitz matrix $\mathbf{T}$ with two 3×3 Toeplitz blocks:

$$\mathbf{T} = \left(\begin{array}{ccc|ccc} 1.00 & 0.30 & 0.05 & 0.20 & 0.10 & 0.00 \\ 0.30 & 1.00 & 0.30 & 0.15 & 0.20 & 0.10 \\ 0.05 & 0.30 & 1.00 & 0.01 & 0.15 & 0.20 \\ \hline 0.20 & 0.15 & 0.01 & 1.00 & 0.30 & 0.05 \\ 0.10 & 0.20 & 0.15 & 0.30 & 1.00 & 0.30 \\ 0.00 & 0.10 & 0.20 & 0.05 & 0.30 & 1.00 \end{array}\right) = \begin{pmatrix} \mathbf{T}^{(1)} & \mathbf{T}^{(2)} \\ \mathbf{T}^{(2)^T} & \mathbf{T}^{(1)} \end{pmatrix} \qquad (2.1)$$

The blocks are

$$\mathbf{T}^{(1)} = \begin{pmatrix} 1.00 & 0.30 & 0.05 \\ 0.30 & 1.00 & 0.30 \\ 0.05 & 0.30 & 1.00 \end{pmatrix} \qquad \mathbf{T}^{(2)} = \begin{pmatrix} 0.20 & 0.10 & 0.00 \\ 0.15 & 0.20 & 0.10 \\ 0.01 & 0.15 & 0.20 \end{pmatrix}$$

Note that $\mathbf{T}^{(1)}$ is symmetric, but $\mathbf{T}^{(2)}$ is not.

We show how to embed $\mathbf{T}$ into a symmetric block circulant matrix $\mathbf{C}$ with circulant blocks. First we embed $\mathbf{T}^{(i)}$'s into circulant $\mathbf{C}^{(i)}$'s:

$$\mathbf{C}^{(1)} = \begin{pmatrix} 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 \\ 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 \\ 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 \\ 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 \\ 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 \\ 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 \end{pmatrix}$$

$$\mathbf{C}^{(2)} = \begin{pmatrix} 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 \\ 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 \\ 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 \\ 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 \\ 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 \\ 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 \end{pmatrix}$$

Next we arrange $\mathbf{C}^{(1)}$ and $\mathbf{C}^{(2)}$ into $\mathbf{C}$:

$$
\mathbf{C} = \left(\begin{array}{cccccc|cccccc|cccccc|cccccc}
1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 \\
0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 \\
0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 \\
0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 \\
0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 \\
0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 \\
\hline
0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\hline
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 \\
\hline
0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 \\
0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 \\
0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 \\
0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 \\
0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 \\
0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00
\end{array}\right)
$$

We have constructed $\mathbf{C}$ – a symmetric $4n_1n_2 \times 4n_1n_2 = 24 \times 24$ block circulant matrix. It has $2n_1 = 4$ circulant blocks of the size $2n_2 \times 2n_2 = 6 \times 6$ each and embeds the original $\mathbf{T}$.

2.4 Storage of Block Circulant Matrices.

One of the important properties of block circulant matrices with circulant blocks is the fact that, similar to the regular circulant matrices, all the information about the matrix is contained in the first column. In particular, any other column can be obtained from the first one as needed. This means that when dealing with a block circulant matrix with circulant blocks, one only has to store its first column rather than the whole matrix.

To illustrate the method, consider again the matrix $\mathbf{C}$ obtained in the previous section. The number of different blocks is $k_1 = 4$, and the block size is $k_2 = 6$. Suppose we want to obtain, say, the ninth column (the numbering of blocks and entries within each block starts at zero).

$$
\mathbf{C} = \left(\begin{array}{c cccccc|cccccc|cccccc|cccccc}
\uparrow & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 0.20 & 0.10 & 0.00 & \mathbf{0.00} & 0.01 & 0.15 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 \\
| & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.15 & 0.20 & 0.10 & \mathbf{0.00} & 0.00 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 \\
| & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & \mathbf{0.10} & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 \\
\perp & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & \mathbf{0.20} & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 \\
\uparrow & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.00 & 0.00 & 0.01 & \mathbf{0.15} & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 \\
\perp & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.10 & 0.00 & 0.00 & \mathbf{0.01} & 0.15 & 0.20 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 \\
\hline
\uparrow & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 1.00 & 0.30 & 0.05 & \mathbf{0.00} & 0.05 & 0.30 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
| & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.30 & 1.00 & 0.30 & \mathbf{0.05} & 0.00 & 0.05 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
| & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & \mathbf{0.30} & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\perp & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & \mathbf{1.00} & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\uparrow & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.05 & 0.00 & 0.05 & \mathbf{0.30} & 1.00 & 0.30 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\perp & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.30 & 0.05 & 0.00 & \mathbf{0.05} & 0.30 & 1.00 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\hline
\uparrow & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.20 & 0.15 & 0.01 & \mathbf{0.00} & 0.00 & 0.10 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 \\
| & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & \mathbf{0.01} & 0.00 & 0.00 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 \\
| & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & \mathbf{0.15} & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 \\
\perp & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & \mathbf{0.20} & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 \\
\uparrow & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & \mathbf{0.10} & 0.20 & 0.15 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 \\
\perp & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.15 & 0.01 & 0.00 & \mathbf{0.00} & 0.10 & 0.20 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 \\
\hline
\uparrow & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.00 & 0.00 & 0.00 & \mathbf{0.00} & 0.00 & 0.00 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 \\
| & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.00 & \mathbf{0.00} & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 \\
| & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & \mathbf{0.00} & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 \\
\perp & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & \mathbf{0.00} & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 \\
\uparrow & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & \mathbf{0.00} & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 \\
\perp & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.00 & 0.00 & 0.00 & \mathbf{0.00} & 0.00 & 0.00 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00
\end{array}\right)
$$

To obtain the ninth column, we must visit blocks of the first column in the following order: 1, 0, 3, 2. In each block, we must read off entries 3, 2, 1, 0, 5, 4.

For a general number of blocks k_1, block size k_2, and target column number i, this transforms to the block ordering from $[i/k_2]$ downward (modulo k_1), and within-block ordering from $i \bmod k_2$ downward (modulo k_2).

2.5 Two-Dimensional Fourier Transform.

There are many definitions of the Fourier transform used in the literature and software packages that differ both in the normalizing constant and the sign in the exponent. We use the following definition of the two-dimensional Fourier transform.

Definition. The two-dimensional Fourier transform of the rectangular $n_1 \times n_2$ matrix $\mathbf{A}$ is the matrix $\mathbf{B}$ of the same size with the entries

$$B_{lm} = \frac{1}{\sqrt{n_1 n_2}} \sum_{g=0}^{n_1-1} \sum_{h=0}^{n_2-1} A_{gh} \exp\left(-\frac{2\pi i g l}{n_1}\right) \exp\left(-\frac{2\pi i h m}{n_2}\right)$$

$$l = 0, \ldots, n_1 - 1, \quad m = 0, \ldots, n_2 - 1$$

When either n_1 or n_2 is one, this reduces to the familiar one-dimensional Fourier transform. From the statistical point of view, this case corresponds to a random field observed over a $n \times 1$ or $1 \times n$ grid, that is, a time series.

The two-dimensional Fourier transform can also be written in the form of matrix multiplication. Consider an $n_1 n_2 \times n_1 n_2$ matrix $\mathbf{F}$ with the entries

$$F_{lm} = \frac{1}{\sqrt{n_1 n_2}} \exp\left(-\frac{2\pi i}{n_1}\left[\frac{l}{n_2}\right]\left[\frac{m}{n_2}\right]\right) \exp\left(-\frac{2\pi i}{n_2}(l \bmod n_2)(m \bmod n_2)\right)$$

$$l, m = 0, 1, \ldots, n_1 n_2 - 1$$

Then $\mathbf{B}$ is obtained from $\mathbf{A}$ as follows:

- write out the entries of $\mathbf{A}$ in a $n_1 n_2$-vector $\mathbf{a}$ row-wise; for example, if

$$\mathbf{A} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix},$$

then

$$\mathbf{a} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}$$

- compute $\mathbf{b} = \mathbf{F}\mathbf{a}$;
- fill in the $n_1 \times n_2$ matrix $\mathbf{B}$ with the components of $\mathbf{b}$ row-wise;

The above algorithm is not a recipe for computations; it only illustrates the connection between the two-dimensional Fourier transform and the matrix multiplication notation. Computationally, however, the Fourier transform can be implemented much more efficiently. In the one-dimensional case, the Cooley-Tukey [13] FFT (fast Fourier transform) algorithm requires $n \log_2 n$ operations to transform a sequence of the length $n = 2^k$ (the Cooley-Tukey algorithm is most beneficial when the length of the sequence is a power of 2). The equivalent matrix times vector multiplication would require n^2 operations.

The two-dimensional transform can be thought of as a sequence of one-dimensional transforms applied first to each row of $\mathbf{A}$ and then to each column of the resulting matrix (or vice versa). Therefore the total number of operations required for the two-dimensional transform of the $n_1 \times n_2$ matrix is

$$n_1(n_2 \log_2 n_2) + n_2(n_1 \log_2 n_1) = n_1 n_2 \log_2(n_1 n_2)$$

which is smaller than $(n_1 n_2)^2$ – the number of operations involved in the $\mathbf{Ma}$ multiplication for a general $n_1 n_2 \times n_1 n_2$ matrix $\mathbf{M}$.

Accordingly, we aim to reduce all the computations to multiplications of the form $\mathbf{Fa}$, and perform this multiplication by the two-dimensional FFT. Namely, to compute $\mathbf{b} = \mathbf{Fa}$, we

- arrange the entries of $\mathbf{a}$ in a $n_1 \times n_2$ matrix $\mathbf{A}$ row-wise;
- compute $\mathbf{B} = \mathrm{FFT}(\mathbf{A})$ by Cooley-Tukey algorithm;

- write out the elements of the matrix **B** in the vector **b** row-wise; then **b** is the desired **Fa**.

The inverse Fourier transform is defined similarly without the minuses in the exponents:

Definition. The two-dimensional inverse Fourier transform of the rectangular $n_1 \times n_2$ matrix **B** is the matrix **A** of the same size with the entries

$$A_{lm} = \frac{1}{\sqrt{n_1 n_2}} \sum_{g=0}^{n_1-1} \sum_{h=0}^{n_2-1} B_{gh} \exp\left(\frac{2\pi igl}{n_1}\right) \exp\left(\frac{2\pi ihm}{n_2}\right)$$

$$l = 0, \ldots, n_1 - 1, \quad m = 0, \ldots, n_2 - 1$$

The same matrix multiplication approach is applicable, with the matrix $\mathbf{F}^H$ instead of $\mathbf{F}$, where

$$\mathbf{F}^H \equiv \bar{\mathbf{F}}^T$$

The normalizing constant $1/\sqrt{n_1 n_2}$ is chosen in such a way that $\mathbf{F}$ is unitary, that is,

$$\mathbf{F}^H = \mathbf{F}^{-1}$$

This also means that if we apply the Fourier transform to **A** and then apply the inverse Fourier transform to the result, we get **A** back.

2.6 Diagonalization of Block Circulant Matrices with Circulant Blocks.

We discussed the two-dimensional Fourier transform to cite the following theorem, which for us is the most important property of block circulant matrices.

Theorem 1 *(Nott, Wilson [47]). For a symmetric block circulant $n_1 n_2 \times n_1 n_2$ matrix $\mathbf{C}$ with n_1 circulant blocks, each of size $n_2 \times n_2$,*

$$\mathbf{C} = \mathbf{F}^H \mathbf{\Lambda} \mathbf{F},$$

where $\mathbf{F}$ is the matrix of the two-dimensional $n_1 \times n_2$ Fourier transform, and $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues of $\mathbf{C}$.

This result gives a practical way of computing the eigenvalues of $\mathbf{C}$. If we multiply both sides by $\mathbf{F}$, we get

$$\mathbf{FC} = \mathbf{\Lambda F}.$$

Consider in this equation first columns only:

$$\mathbf{FC}_1 = \mathbf{\Lambda F}_1,$$

and note that the first column of $\mathbf{F}$ consists entirely of values $1/\sqrt{n_1 n_2}$. Therefore,

$$\mathbf{FC}_1 = \frac{1}{\sqrt{n_1 n_2}} \boldsymbol{\lambda}$$

and

$$\boldsymbol{\lambda} = \sqrt{n_1 n_2}\, \mathbf{FC}_1.$$

Hence we have the following algorithm for computing $\boldsymbol{\lambda}$, the vector of eigenvalues of $\mathbf{C}$:

- assemble $\mathbf{C}_1$ (the first column of $\mathbf{C}$) in a rectangular $n_1 \times n_2$ matrix row-wise;
- take the two-dimensional Fourier transform of this matrix;

- multiply the resulting matrix by $\sqrt{n_1 n_2}$;
- read off the product row-wise in a $n_1 n_2$ -vector $\boldsymbol{\lambda}$.

Note that this is in line with our earlier observation that all the information about $\mathbf{C}$ is contained in its first column $\mathbf{C}_1$.

To continue with our example, consider again the matrix

$$\mathbf{C} = \left(\begin{array}{cccccc|cccccc|cccccc|cccccc}
1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 \\
0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 \\
0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 \\
0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 \\
0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 \\
0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 \\
\hline
0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
\hline
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 \\
\hline
0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 \\
0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 \\
0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 \\
0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 \\
0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 \\
0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00
\end{array}\right)$$

To find its eigenvalues, we arrange the elements of the first column of $\mathbf{C}$ into a 4×6 matrix row-wise, apply to it the two-dimensional Fourier transform, and multiply the result by $\sqrt{24}$:

$$\mathbf{\Lambda} = \sqrt{24}\,\mathrm{FFT}\begin{pmatrix} 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 \\ 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 \end{pmatrix} =$$

$$= \begin{pmatrix} 2.620000 & 1.890000 & 0.790000 & 0.420000 & 0.790000 & 1.890000 \\ 1.700000 & 1.353923 & 0.719282 & 0.150000 & 0.580718 & 1.146077 \\ 0.780000 & 0.610000 & 0.510000 & 0.580000 & 0.510000 & 0.610000 \\ 1.700000 & 1.146077 & 0.580718 & 0.500000 & 0.719282 & 1.353923 \end{pmatrix}$$

These are the 24 eigenvalues of $\mathbf{C}$.

2.7 Computations with block circulant matrices.

2.7.1 Determinant.

Once we know the eigenvalues of $\mathbf{C}$, we can easily compute the determinant. Since $\mathbf{F}$ is unitary, $|\mathbf{F}| = |\mathbf{F}^H| = 1$, so

$$|\mathbf{C}| = |\mathbf{F}^H\mathbf{\Lambda}\mathbf{F}| = |\mathbf{F}^H||\mathbf{\Lambda}||\mathbf{F}| = |\mathbf{\Lambda}| = \prod_{i=1}^{n_1n_2} \lambda_i$$

2.7.2 Quadratic Form.

Suppose that for an arbitrary real-valued n_1n_2-vector $\mathbf{x}$ we want to compute the quadratic form $\mathbf{x}^T\mathbf{C}\mathbf{x}$. Using the diagonalization theorem, we reduce this to another application of the FFT:

$$\begin{aligned}\mathbf{x}^T\mathbf{C}\mathbf{x} = \mathbf{x}^T(\mathbf{F}^H\boldsymbol{\Lambda}\mathbf{F})\mathbf{x} = (\mathbf{F}\mathbf{x})^H\boldsymbol{\Lambda}(\mathbf{F}\mathbf{x}) = \mathbf{y}^H\boldsymbol{\Lambda}\mathbf{y} = \\ = \sum_{i=1}^{n_1n_2}|y_i|^2\lambda_i.\end{aligned}$$

Therefore, the algorithm for computing $\mathbf{x}^T\mathbf{C}\mathbf{x}$ is as follows:

- compute $\boldsymbol{\lambda}$;
- assemble $\mathbf{x}$ in a rectangular $n_1 \times n_2$ matrix $\mathbf{X}$ row-wise;
- take the two-dimensional Fourier transform of $\mathbf{X}$, and call the complex-valued result $\mathbf{Y}$;
- read off $\mathbf{Y}$ row-wise in a complex n_1n_2 -vector $\boldsymbol{y}$;
- $\mathbf{x}^T\mathbf{C}\mathbf{x} = \sum_{i=1}^{n_1n_2}|y_i|^2\lambda_i$.

2.7.3 The Inverse

Now suppose that we want to invert $\mathbf{C}$. We use the fact that the inverse of the symmetric block circulant matrix with circulant blocks is also symmetric block circulant with circulant blocks (Trapp [61], Theorem 6). Therefore it suffices to find the first column of $\mathbf{C}^{-1}$ only. The process is analogous to finding the eigenvalues:

$$\mathbf{C}^{-1} = (\mathbf{F}^H\boldsymbol{\Lambda}\mathbf{F})^{-1} = \mathbf{F}^H\boldsymbol{\Lambda}^{-1}\mathbf{F} \quad (\text{since } \mathbf{F}^H = \mathbf{F}^{-1}).$$

After multiplying both sides by $\mathbf{F}$,

$$\mathbf{F}\mathbf{C}^{-1} = \boldsymbol{\Lambda}^{-1}\mathbf{F}.$$

Considering the first columns only,

$$\mathbf{F}(\mathbf{C}^{-1})_1 = \frac{1}{\sqrt{n_1 n_2}}(1/\lambda_1, 1/\lambda_2, \dots, 1/\lambda_{n_1 n_2})^T.$$

Finally, multiplying both sides by $\mathbf{F}^H$,

$$(\mathbf{C}^{-1})_1 = \mathbf{F}^H \left(\frac{1}{\sqrt{n_1 n_2}}(1/\lambda_1, 1/\lambda_2, \dots, 1/\lambda_{n_1 n_2})^T \right).$$

Hence we have the following algorithm for obtaining the first column of $\mathbf{C}^{-1}$:

- compute $\boldsymbol{\lambda}$;
- invert each component of $\boldsymbol{\lambda}$ and divide it by $\sqrt{n_1 n_2}$;
- assemble the result in a rectangular $n_1 \times n_2$ matrix and take the two-dimensional inverse Fourier transform;
- read off the result of the transform row-wise in a $n_1 n_2$-vector $(\mathbf{C}^{-1})_1$

2.8 Implementation details.

For actual simulations we use a version of FFT by Frigo and Johnson, called FFTW (which stands for "Fastest Fourier Transform in the West"). As Frigo and Johnson write in the introduction to [26],

> "In the past, speed was the direct consequence of clever algorithms that minimized the number of arithmetic operations. On present-day general-purpose microcomputers, however, the performance of a program is mostly determined by complicated interactions of the code with the processor pipeline, and by the structure of the memory."

The FFTW is a C library implementing the Cooley-Tukey algorithm [13] for an n-dimensional Fourier transform designed with this paradigm in mind. It is publicly available at WWW site [26]. According to the experiments of its authors, the library typically yields significantly better performance than all other publicly available DFT software, and, while retaining complete portability, is competitive with or faster than proprietary codes such as Sun's Performance Library and IBM's ESSL library that are highly tuned for a single machine.

We note two details of FFTW important for this dissertation. First, its definition of the forward and the backward transform differs from ours in lacking the normalizing constants in both cases. In particular, this means that applying the forward and then the backward FFTW will multiply the input by $n_1 n_2$.

The second comment regards the allocation of the two-dimensional arrays in C. Our algorithms require frequent conversions from $n_1 n_2$-vectors to $n_1 \times n_2$ matrices filled with the entries of those vectors row-wise. If these conversions where taken literally as copying the entries from one memory location to another, this would constitute a huge overhead on the algorithms. However, one of the C memory models, consistent with the FFTW memory usage, is such that the two-dimensional arrays *are* in fact long one-dimensional arrays stored row-wise (this is also known as C row-major order as opposed to the FORTRAN column-major order). Specifically, if we set

```
c = (double *) malloc(n1 * n2 * sizeof(double));
```

we can treat `c` simultaneously as a long vector and a rectangular matrix obtained from it row-wise, accessing C_{ij} as `c[i*n2 + j]`, so that no additional transformation is necessary.

Chapter 3

GENERATION OF GAUSSIAN RANDOM FIELDS

3.1 Introduction

In this chapter we discuss algorithms for generating stationary Gaussian random fields over regular grids, either unconditionally or given that certain (in particular, all) field values fall within specified intervals. Since many statistical models for spatial processes are based on transformations of Gaussian fields, being able to generate such fields is important for doing simulations and studying model properties. Conditional generation is required, among other applications, for the EM algorithm in discrete models in setups where the Gaussian field plays the role of the unobserved data (see next Chapter).

Neither generation problem is simple, particularly when the grid size is large.

3.2 Gaussian Random Fields

A *stochastic process*[1] is a collection of random variables

$$\{Z(\mathbf{s}) : \mathbf{s} \in D\}$$

indexed over a set D.

The Daniel-Kolmogorov theorem states that to specify a stochastic process all we have to do is to give the joint distribution of any finite subset $\{Z(\mathbf{s}_1), \ldots, Z(\mathbf{s}_n)\}$ in a consistent way, that is, subject to a condition that for $\mathbf{t} \neq \mathbf{s}_1, \ldots, \mathbf{s}_m$,

$$P(Z(\mathbf{s}_i) \in A_i, i = 1, \ldots, m, Z(\mathbf{t}) \in \mathbb{R}) = P(Z(\mathbf{s}_i) \in A_i, i = 1, \ldots, m)$$

In this dissertation we consider *random fields*. A random field is a particular stochastic process where the index set D is a subset of $\mathbb{R}^2$.

We say that a random field is *(strictly) stationary* if its distribution is unchanged when the origin of the index set is translated. If the distribution is also unchanged when the index set is rotated about the origin, the field is called *isotropic*.

Another, weaker type of stationarity is the second-order stationarity, defined through the covariance function. In general, for every random field $\mathbf{Z}$ we introduce the *mean function*

$$m(\mathbf{s}) = E(Z(\mathbf{s}))$$

and the *covariance function*

$$r(\mathbf{s}, \mathbf{t}) = \mathrm{Cov}(Z(\mathbf{s}), Z(\mathbf{t}))$$

[1]General theory on stochastic processes appears in Adler [1], Matérn [43], and Yaglom [65].

If $m(\mathbf{s}) \equiv \mu$ and if r is a function of $\mathbf{s}-\mathbf{t}$ only, $\mathbf{Z}$ is called *second-order stationary.* Similarly, if $m(\mathbf{s}) \equiv \mu$ and if r depends only on $\|\mathbf{s}-\mathbf{t}\|$, $\mathbf{Z}$ is called *second-order isotropic.*

In this chapter we consider *Gaussian* fields, defined by the property that all finite collections $(Z(\mathbf{s}_1), \ldots, Z(\mathbf{s}_n))$ are jointly normal. The distribution of such a field is completely determined by its mean and covariance functions. Therefore in this case strict- and second-order properties coincide.

Any function m could be a mean function for some Gaussian process, while necessary and sufficient conditions on r are symmetry and *non-negative definiteness*:

$$r(\mathbf{s}_1, \mathbf{s}_2) = r(\mathbf{s}_2, \mathbf{s}_1)$$
$$\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j r(\mathbf{s}_i, \mathbf{s}_j) = E\left(\sum_{i=1}^{n} \alpha_i Z(\mathbf{s}_i)\right)^2 \geq 0$$

for all n, $\alpha_1, \ldots, \alpha_n, \mathbf{s}_1, \ldots, \mathbf{s}_n$ (Breiman, [7], Chapter 11). In this chapter we are interested in stationary Gaussian fields with $m(\mathbf{s}) \equiv 0$.

3.3 Unconditional Generation

3.3.1 Old Methods

Suppose that we want to generate a sample from a zero-mean Gaussian field with a given covariance function r. To specify the problem completely, we must choose a finite subset $S = \{\mathbf{s}_1, \ldots, \mathbf{s}_n\} \subset \mathbb{R}^2$ for which the values $\mathbf{Z}(\mathbf{s}_i)$ will be generated.

For small values of n and positive-definite covariance functions, the straightforward Cholesky decomposition method (Cressie [15], page 201) can be used. First, we construct a $n \times n$ symmetric positive-definite covariance matrix $\mathbf{T}$ with entries

$$T_{ij} = \mathrm{Cov}(Z(\mathbf{s}_i), Z(\mathbf{s}_j)) = r(\mathbf{s}_i, \mathbf{s}_j),$$

and the problem reduces to generating a multivariate normal vector

$$(Z(\mathbf{s}_1), \ldots, Z(\mathbf{s}_n)) \sim N(\mathbf{0}, \mathbf{T}).$$

(The properties of $\mathbf{T}$ follow from the corresponding properties of r.) This is done by computing the square root of the matrix $\mathbf{T}$ by Cholesky decomposition (Press et al. [51], Chapter 2.9):

$$\mathbf{T} = \mathbf{L}\mathbf{L}^T,$$

where $\mathbf{L}$ is a lower triangular matrix. This decomposition requires $n^3/6$ multiplications and taking n square roots (which is about a factor 2 better than the $\mathbf{LU}$ decomposition in which the symmetry of $\mathbf{T}$ would be ignored). Once we know $\mathbf{L}$, $\mathbf{Z}$ is generated by

$$\mathbf{Z} = \mathbf{L}\boldsymbol{\varepsilon},$$

where

$$\varepsilon_1, \ldots, \varepsilon_n \sim N(0,1) \quad \text{i.i.d.}$$

Indeed, for $\mathbf{Z}$ defined in this way,

$$\begin{aligned} E(\mathbf{Z}) &= E(\mathbf{L}\boldsymbol{\varepsilon}) = \mathbf{L}E(\boldsymbol{\varepsilon}) = \mathbf{0} \\ \mathrm{Var}(\mathbf{Z}) &= \mathrm{Var}(\mathbf{L}\boldsymbol{\varepsilon}) = \mathbf{L}\mathrm{Var}(\boldsymbol{\varepsilon})\mathbf{L}^T = \mathbf{T} \end{aligned}$$

The downside of this simple and powerful method is the fact that to generate a vector $\mathbf{Z}$ of length n we have to deal with the $n \times n$ matrix $\mathbf{T}$. For large values of n the decomposition of $\mathbf{T}$ becomes too expensive, and for even larger n's it becomes impossible even to store $\mathbf{T}$.

If we assume stationarity, the so-called spectral (Shinozuka and Jan, [55], Mejia and Rodrigez-Iturbe [45], Borgman et al. [6]) and turning-bands (Mantoglou and Wilson [40], Tompson et al. [60]) methods become available. Realizations based on the spectral method are obtained by summing a finite cosine series whose coefficients have uniformly distributed phases and amplitudes proportional to the spectral density function of the process. This summation can be done by the FFT, so the cost is about $n \log_2 n$ operations. In the turning bands approach, fields are generated from appropriately summed line processes. The cost of this method when l lines of p nodes each are used is $nl + lc(p)$, where $c(p)$ is the cost of generating the single line realization. Both methods have much more modest computational requirements than the Cholesky method, at the price of being approximate.

In recent years, a new method for generating Gaussian fields was developed, based on circulant embedding (Dembo et al. [18], Davies and Harte [17], Dietrich and Newsam [22], Wood and Chan [63]). It has its limitations: the covariance must fall off rapidly, and the grid should be large enough so that the covariance between points on opposite sides of the grid is negligible. However, within these limitations the method is both exact and computationally as inexpensive as the turning bands or FFT method. Therefore the circulant embedding method is our method of choice; we describe it in detail in the next section.

3.3.2 Circulant Embedding Method

Suppose that we want to generate a sample from a zero-mean Gaussian field $\mathbf{Z}$ with a given covariance function r over the $n_1 \times n_2$ grid $\mathbf{S} = \{\mathbf{s}_{kj}\}$. The grid is assumed regular, with fixed steps x and y:

$$\mathbf{s}_{kj} = \mathbf{s}_{00} + ky\mathbf{v}_2 + jx\mathbf{v}_1, \quad k = 0, \dots, n_1 - 1, \quad j = 0, \dots, n_2 - 1,$$
$$\text{where} \quad \mathbf{v}_1 = (1, 0), \mathbf{v}_2 = (0, 1).$$

Note that grid locations are numbered starting at zero. We write Z_{kj} for the value of $\mathbf{Z}$ at the site $\mathbf{s}_{kj}$. For the purposes of the method, however, it is more convenient to assemble the values of $\mathbf{Z}$ into a n_1n_2-vector, and switch to the notation Z_i for the value of $\mathbf{Z}$ at the i-th location of the grid enumerated row-wise:

$$Z_i \equiv Z_{[i/n_2], i\text{mod}n_2}, \quad i = 0, ..., n_1n_2 - 1$$

The first observation is that under the assumption of regularity of the grid $\mathbf{S}$ and stationarity of the field, the $n_1n_2 \times n_1n_2$ covariance matrix $\mathbf{T}$ of the n_1n_2-vector $\mathbf{Z}$ is block Toeplitz with Toeplitz blocks (Zimmerman [66]). Namely, it consists of n_1 different Toeplitz blocks, each of size $n_2 \times n_2$:

$$\mathbf{T} = \begin{pmatrix} \mathbf{T}^{(1)} & \mathbf{T}^{(2)} & \dots & \dots & \mathbf{T}^{(n_1)} \\ \mathbf{T}^{(2)^T} & \ddots & \ddots & & \\ \mathbf{T}^{(3)^T} & \ddots & \ddots & \ddots & \\ \vdots & \ddots & \ddots & \ddots & \\ \mathbf{T}^{(n_1)^T} & & & & \ddots \end{pmatrix}$$

Once we recognize the structure of $\mathbf{T}$, we recall that, as established in the previous chapter, symmetric block Toeplitz matrices with Toeplitz blocks should be embedded into larger symmetric block circulant matrices with circulant blocks for faster computations. So, we move on to the $4n_1n_2 \times 4n_1n_2$ matrix $\mathbf{C}$ which embeds $\mathbf{T}$, using the standard embedding procedure (see Section 2.3).

To continue, we need to assume that $\mathbf{C}$ is non-negative definite. Otherwise, the algorithm fails, which indicates that the grid is too small or that r does not fall off rapidly enough with distance. In this case we need to back up and increase the grid size so that the covariance between the points on the opposite sides of the grid is nearly zero. Exact results on sufficient conditions for $\mathbf{C}$ to be non-negative definite can be found in Dietrich and Newsam [23].

Let's assume, however, that $\mathbf{C}$ is positive definite (we will have a chance to verify that later). Now the idea is to generate a $4n_1n_2$-vector

$$\mathbf{W} \sim N(\mathbf{0}, \mathbf{C})$$

and then extract certain components of $\mathbf{W}$, which, when assembled into a n_1n_2-vector $\mathbf{Z}$, will be jointly normal with the required parameters.

First we give the algorithm to generate $\mathbf{W}$.

For the two-dimensional $2n_1 \times 2n_2$ Fourier transform matrix $\mathbf{F}$, define

$$\mathbf{F}_1 = \operatorname{Re} \mathbf{F}$$

$$\mathbf{F}_2 = \operatorname{Im} \mathbf{F}$$

Then, by the diagonalization theorem,

$$\begin{aligned}\mathbf{C} &= \mathbf{F}^H \boldsymbol{\Lambda} \mathbf{F} \\ &= (\mathbf{F}_1 + i\mathbf{F}_2)^H \boldsymbol{\Lambda} (\mathbf{F}_1 + i\mathbf{F}_2) \\ &= (\mathbf{F}_1 - i\mathbf{F}_2)^T \boldsymbol{\Lambda} (\mathbf{F}_1 + i\mathbf{F}_2) \\ &= (\mathbf{F}_1 - i\mathbf{F}_2) \boldsymbol{\Lambda} (\mathbf{F}_1 + i\mathbf{F}_2) \quad (\mathbf{F} \text{ is symmetric, hence so are } \mathbf{F}_1 \text{ and } \mathbf{F}_2) \\ &= (\mathbf{F}_1 \boldsymbol{\Lambda} \mathbf{F}_1 + \mathbf{F}_2 \boldsymbol{\Lambda} \mathbf{F}_2) + i(\mathbf{F}_1 \boldsymbol{\Lambda} \mathbf{F}_2 - \mathbf{F}_2 \boldsymbol{\Lambda} \mathbf{F}_1).\end{aligned}$$

Equating real and imaginary parts (in fact, $\mathbf{C}$ is real), we get

$$\mathbf{F}_1 \boldsymbol{\Lambda} \mathbf{F}_1 + \mathbf{F}_2 \boldsymbol{\Lambda} \mathbf{F}_2 = \mathbf{C} \tag{3.1}$$

$$\mathbf{F}_1 \boldsymbol{\Lambda} \mathbf{F}_2 - \mathbf{F}_2 \boldsymbol{\Lambda} \mathbf{F}_1 = \mathbf{0} \tag{3.2}$$

Now take two independent vectors $\mathbf{e}_1$ and $\mathbf{e}_2$ with independent components

$$\mathbf{e}_1, \mathbf{e}_2 \sim N(\mathbf{0}, \boldsymbol{\Lambda})$$

and combine them into a complex vector $\mathbf{e} = \mathbf{e}_1 + i\mathbf{e}_2$. Recall that $\boldsymbol{\lambda}$ – the vector of eigenvalues needed to build $\boldsymbol{\Lambda}$ – is easily computed by the two-dimensional FFT of the $2n_1 \times 2n_2$ matrix obtained from the first column of $\mathbf{C}$. This is the point where we check the non-negative definiteness of the $\mathbf{C}$: all the λ_i's are supposed to be non-negative.

Define

$$\begin{aligned}\mathbf{w} &= \mathbf{F}\mathbf{e} \\ &= (\mathbf{F}_1 + i\mathbf{F}_2)(\mathbf{e}_1 + i\mathbf{e}_2) \\ &= (\mathbf{F}_1\mathbf{e}_1 - \mathbf{F}_2\mathbf{e}_2) + i(\mathbf{F}_2\mathbf{e}_1 + \mathbf{F}_1\mathbf{e}_2) \\ &\equiv \mathbf{w}_1 + i\mathbf{w}_2.\end{aligned}$$

Note again that though we write $\mathbf{w} = \mathbf{F}\mathbf{e}$, the actual computation is, in fact, the two-dimensional FFT of the $\mathbf{e}$ re-arranged into a matrix.

We will now show that $\mathbf{w}_1$ and $\mathbf{w}_2$ are independent realizations of $N(\mathbf{0}, \mathbf{C})$. Obviously they are zero mean normal, so we need only look at the covariance matrix:

$$\begin{aligned}
\mathrm{E}(\mathbf{w}_1\mathbf{w}_1^T) &= \mathrm{E}(\mathbf{F}_1\mathbf{e}_1 - \mathbf{F}_2\mathbf{e}_2)(\mathbf{F}_1\mathbf{e}_1 - \mathbf{F}_2\mathbf{e}_2)^T \\
&= \mathrm{E}(\mathbf{F}_1\mathbf{e}_1 - \mathbf{F}_2\mathbf{e}_2)(\mathbf{e}_1^T\mathbf{F}_1 - \mathbf{e}_2^T\mathbf{F}_2) \\
&= \mathrm{E}(\mathbf{F}_1\mathbf{e}_1\mathbf{e}_1^T\mathbf{F}_1 - \mathbf{F}_2\mathbf{e}_2\mathbf{e}_1^T\mathbf{F}_1 - \mathbf{F}_1\mathbf{e}_1\mathbf{e}_2^T\mathbf{F}_2 + \mathbf{F}_2\mathbf{e}_2\mathbf{e}_2^T\mathbf{F}_2) \\
&= \mathbf{F}_1\mathrm{E}(\mathbf{e}_1\mathbf{e}_1^T)\mathbf{F}_1 - \mathbf{F}_2\mathrm{E}(\mathbf{e}_2\mathbf{e}_1^T)\mathbf{F}_1 - \mathbf{F}_1\mathrm{E}(\mathbf{e}_1\mathbf{e}_2^T)\mathbf{F}_2 + \mathbf{F}_2\mathrm{E}(\mathbf{e}_2\mathbf{e}_2^T)\mathbf{F}_2 \\
&= \mathbf{F}_1\mathbf{\Lambda}\mathbf{F}_1 - \mathbf{F}_2\mathbf{0}\mathbf{F}_1 - \mathbf{F}_1\mathbf{0}\mathbf{F}_2 + \mathbf{F}_2\mathbf{\Lambda}\mathbf{F}_2 \\
&= \mathbf{F}_1\mathbf{\Lambda}\mathbf{F}_1 + \mathbf{F}_2\mathbf{\Lambda}\mathbf{F}_2 \\
&= \mathbf{C} \quad \text{(because of (3.1))}.
\end{aligned}$$

In the same fashion we can show that

$$\mathrm{E}\mathbf{w}_2\mathbf{w}_2^T = \mathbf{C}.$$

To prove independence of $\mathbf{w}_1$ and $\mathbf{w}_2$ observe that

$$\begin{aligned}
\mathrm{E}(\mathbf{w}_1\mathbf{w}_2^T) &= \mathrm{E}(\mathbf{F}_1\mathbf{e}_1 - \mathbf{F}_2\mathbf{e}_2)(\mathbf{F}_2\mathbf{e}_1 + \mathbf{F}_1\mathbf{e}_2)^T \\
&= \mathrm{E}(\mathbf{F}_1\mathbf{e}_1 - \mathbf{F}_2\mathbf{e}_2)(\mathbf{e}_1^T\mathbf{F}_2 + \mathbf{e}_2^T\mathbf{F}_1) \\
&= \mathrm{E}(\mathbf{F}_1\mathbf{e}_1\mathbf{e}_1^T\mathbf{F}_2 - \mathbf{F}_2\mathbf{e}_2\mathbf{e}_1^T\mathbf{F}_2 + \mathbf{F}_1\mathbf{e}_1\mathbf{e}_2^T\mathbf{F}_1 - \mathbf{F}_2\mathbf{e}_2\mathbf{e}_2^T\mathbf{F}_1) \\
&= \mathbf{F}_1\mathrm{E}(\mathbf{e}_1\mathbf{e}_1^T)\mathbf{F}_2 - \mathbf{F}_2\mathrm{E}(\mathbf{e}_2\mathbf{e}_1^T)\mathbf{F}_2 + \mathbf{F}_1\mathrm{E}(\mathbf{e}_1\mathbf{e}_2^T)\mathbf{F}_1 - \mathbf{F}_2\mathrm{E}(\mathbf{e}_2\mathbf{e}_2^T)\mathbf{F}_1 \\
&= \mathbf{F}_1\mathbf{\Lambda}\mathbf{F}_2 - \mathbf{F}_2\mathbf{0}\mathbf{F}_2 + \mathbf{F}_1\mathbf{0}\mathbf{F}_1 + \mathbf{F}_2\mathbf{\Lambda}\mathbf{F}_1 \\
&= \mathbf{F}_1\mathbf{\Lambda}\mathbf{F}_2 - \mathbf{F}_2\mathbf{\Lambda}\mathbf{F}_1 \\
&= \mathbf{0} \quad \text{(because of (3.2))}.
\end{aligned}$$

which, together with normality, gives independence.

Now that we have a realization of

$$\mathbf{W} \sim N(\mathbf{0}, \mathbf{C}),$$

(in fact, we have two independent realizations), we show how to recover

$$\mathbf{Z} \sim N(\mathbf{0}, \mathbf{T}),$$

which is what we really need.

Recall how the matrix $\mathbf{C}$ looks in terms of the original $\mathbf{T}^{(i)}$'s that make up the $\mathbf{T}$:

$$\mathbf{C} = \left(\begin{array}{cccccc|cc}
\mathbf{T}^{(1)} & \dots & \mathbf{T}^{(2)} & \dots & \mathbf{T}^{(3)} & \dots & \dots & \dots \\
\dots & \mathbf{T}^{(1)} & \dots & \mathbf{T}^{(2)} & \dots & \mathbf{T}^{(3)} & \dots & \dots \\
 & & & & & & & \\
{\mathbf{T}^{(2)}}^T & \dots & \mathbf{T}^{(1)} & \dots & \mathbf{T}^{(2)} & \dots & \dots & \dots \\
\dots & {\mathbf{T}^{(2)}}^T & \dots & \mathbf{T}^{(1)} & \dots & \mathbf{T}^{(2)} & \dots & \dots \\
 & & & & & & & \\
{\mathbf{T}^{(3)}}^T & \dots & {\mathbf{T}^{(2)}}^T & \dots & \mathbf{T}^{(1)} & \dots & \dots & \dots \\
\dots & {\mathbf{T}^{(3)}}^T & \dots & {\mathbf{T}^{(2)}}^T & \dots & \mathbf{T}^{(1)} & \dots & \dots \\
\hline
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots
\end{array}\right)$$

We see that one should take the first half of $\mathbf{W}$, divide it into groups of length n_2, and take every other group:

$$\begin{aligned}
\mathbf{Z} = (&W_0, W_1, \dots, W_{n_2-1}, \\
&W_{2n_2}, W_{2n_2+1}, \dots, W_{3n_2-1}, \\
&\dots, \dots, \dots, \dots, \\
&W_{2(n_1-1)n_2}, W_{2(n_1-1)n_2+1}, \dots, W_{2(n_1-1)n_2+n_2-1})
\end{aligned}$$

Then $Z \sim N(\mathbf{0}, \mathbf{T})$.

We go over the algorithm for generating $\mathbf{Z}$, counting operations.

- Obtain the first column of $\mathbf{C}$ from the given $\mathbf{T}$ and arrange it into a $2n_1 \times 2n_2$ matrix row-wise.

- Take the two-dimensional FFT of this matrix to get $\boldsymbol{\lambda}$, the vector of eigenvalues of $\mathbf{C}$, at a cost of $4n_1n_2 \log_2(4n_1n_2)$ operations.

- Check whether each λ_j is non-negative; if so, continue. Otherwise the algorithm fails (increase n_1 and n_2 and start over).

- For each $j = 1, \ldots, 4n_1n_2$, generate $\varepsilon_1, \varepsilon_2 \sim N(0, \lambda_j)$, combine them in a complex vector $\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}_1 + i\boldsymbol{\varepsilon}_2$ and arrange it into a $2n_1 \times 2n_2$ matrix row-wise;

- Take the two-dimensional FFT of this matrix to get complex vector $\mathbf{w}$ at a cost of $4n_1n_2 \log_2(4n_1n_2)$ operations.

- Consider the real and imaginary parts of $\mathbf{w}$ separately and extract the useful components from each of the two as described above; these are independent $\mathbf{z}_1$, $\mathbf{z}_2 \sim N(\mathbf{0}, \mathbf{T})$.

Therefore the total cost for generating two independent observations of $\mathbf{Z}$ is about $8n_1n_2 \log_2(4n_1n_2)$, with a storage requirement of $4n_1n_2$ complex numbers. Both requirements show the overwhelming advantage of this method over the Cholesky decomposition, as seen in Table 3.1.

Note also that the circulant embedding method produces not just one, but two independent realizations of $\mathbf{Z}$.

We conclude this section with an example.

	Cholesky decomposition	Circulant embedding
"diagonalization" of the covariance matrix	$(n_1 n_2)^3/6$ multiplications and $n_1 n_2$ square roots	$4n_1 n_2 \log_2(4n_1 n_2)$ multiplications
generation of the field	$n_1 n_2(n_1 n_2 + 1)/2$ multiplications (for $\mathbf{L}\boldsymbol{\varepsilon}$)	$4n_1 n_2 \log_2(4n_1 n_2)$ multiplications
storage	$(n_1 n_2)^2$ real values	$4n_1 n_2$ complex values

Table 3.1: Computational requirements for Cholesky decomposition and circulant embedding methods.

Let $n_1 = 2, n_2 = 3$, and consider a 2×3 grid with stationary (yet nonisotropic) covariance structure described in Figure 3.1.

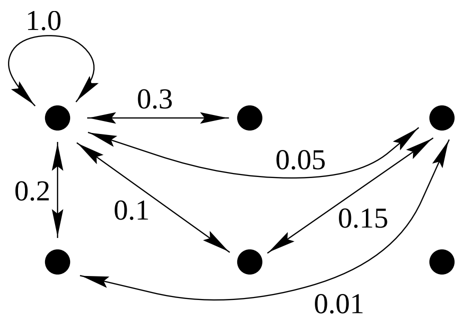

Figure 3.1: Covariance structure used in the example.

If we number our six points in the English reading order, the covariance matrix $\mathbf{T}$ is block Toeplitz with two 3×3 Toeplitz blocks:

$$
\begin{aligned}
\mathbf{T} &= \{\mathrm{Cov}(Z_i, Z_j)\}_{i,j=1,\ldots,6} \\
&= \left(\begin{array}{ccc|ccc}
1.00 & 0.30 & 0.05 & 0.20 & 0.10 & 0.00 \\
0.30 & 1.00 & 0.30 & 0.15 & 0.20 & 0.10 \\
0.05 & 0.30 & 1.00 & 0.01 & 0.15 & 0.20 \\
\hline
0.20 & 0.15 & 0.01 & 1.00 & 0.30 & 0.05 \\
0.10 & 0.20 & 0.15 & 0.30 & 1.00 & 0.30 \\
0.00 & 0.10 & 0.20 & 0.05 & 0.30 & 1.00
\end{array}\right) \\
&= \begin{pmatrix} \mathbf{T}^{(1)} & \mathbf{T}^{(2)} \\ \mathbf{T}^{(2)^T} & \mathbf{T}^{(1)} \end{pmatrix}
\end{aligned}
$$

This is the matrix $\mathbf{T}$ (see (2.1)) from our numerical example in the previous chapter. We proceed in the way described there, embedding it into $\mathbf{C}$ and obtaining its 24 eigenvalues:

$$\boldsymbol{\lambda} = \begin{pmatrix} 2.62000 & 1.89000 & 0.79000 & 0.42000 & 0.79000 & 1.89000 \\ 1.70000 & 1.35392 & 0.71928 & 0.15000 & 0.58072 & 1.14608 \\ 0.78000 & 0.61000 & 0.51000 & 0.58000 & 0.51000 & 0.61000 \\ 1.70000 & 1.14608 & 0.58072 & 0.50000 & 0.71928 & 1.35392 \end{pmatrix}.$$

We see that they are all positive, so $\mathbf{C}$ is indeed positive definite and the method is applicable.

Next we generate 48 mutually independent zero-mean normal variables with variances λ_i (two independent realizations for each of the 24 λ_i's). They are combined into a complex 4×6 matrix, and the two-dimensional FFT is applied to it:

$$\begin{aligned} &\mathbf{W}_1 + i\mathbf{W}_2 \equiv \\ &\equiv \mathrm{FFT}\left(\begin{pmatrix} N(0,2.62000) & N(0,1.89000) & N(0,0.79000) & N(0,0.42000) & N(0,0.79000) & N(0,1.89000) \\ N(0,1.70000) & N(0,1.35392) & N(0,0.71928) & N(0,0.15000) & N(0,0.58072) & N(0,1.14608) \\ N(0,0.78000) & N(0,0.61000) & N(0,0.51000) & N(0,0.58000) & N(0,0.51000) & N(0,0.61000) \\ N(0,1.70000) & N(0,1.14608) & N(0,0.58072) & N(0,0.50000) & N(0,0.71928) & N(0,1.35392) \end{pmatrix} + \right. \\ &\left. + \ i \begin{pmatrix} N(0,2.62000) & N(0,1.89000) & N(0,0.79000) & N(0,0.42000) & N(0,0.79000) & N(0,1.89000) \\ N(0,1.70000) & N(0,1.35392) & N(0,0.71928) & N(0,0.15000) & N(0,0.58072) & N(0,1.14608) \\ N(0,0.78000) & N(0,0.61000) & N(0,0.51000) & N(0,0.58000) & N(0,0.51000) & N(0,0.61000) \\ N(0,1.70000) & N(0,1.14608) & N(0,0.58072) & N(0,0.50000) & N(0,0.71928) & N(0,1.35392) \end{pmatrix} \right). \end{aligned}$$

The real and imaginary parts of the result are two independent realizations of $N(\mathbf{0}, \mathbf{C})$.

To recover the useful components from $\mathbf{W}_1$ and $\mathbf{W}_2$, look at the matrix $\mathbf{C}$ again and note where the entries of the original $\mathbf{T}$ are:

$$\mathbf{C} = \begin{pmatrix}
1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 \\
0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 \\
0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 \\
0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 \\
0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 \\
0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 \\
0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 \\
0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 \\
0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.00 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 & 0.05 \\
0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 & 0.00 \\
0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.01 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 & 0.05 \\
0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.10 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.15 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00 & 0.30 \\
0.10 & 0.00 & 0.00 & 0.01 & 0.15 & 0.20 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.15 & 0.01 & 0.00 & 0.00 & 0.10 & 0.20 & 0.30 & 0.05 & 0.00 & 0.05 & 0.30 & 1.00
\end{pmatrix}$$

The blocks of $\mathbf{T}$ are the four outlined squares. Hence we see that the components of $\mathbf{W}$ with the numbers $\{0, 1, 2, 6, 7, 8\}$ (numbering starts at zero) have the desired $N(\mathbf{0}, \mathbf{T})$ distribution. Therefore, the two resulting independent samples of the field over our 2×3 grid are

$$\begin{pmatrix} {W_1}^{(0)} & {W_1}^{(1)} & {W_1}^{(2)} \\ {W_1}^{(6)} & {W_1}^{(7)} & {W_1}^{(8)} \end{pmatrix}$$

and

$$\begin{pmatrix} {W_2}^{(0)} & {W_2}^{(1)} & {W_2}^{(2)} \\ {W_2}^{(6)} & {W_2}^{(7)} & {W_2}^{(8)} \end{pmatrix}$$

3.4 Conditional Generation

3.4.1 The problem

As before, we want to generate a sample from the Gaussian random field, but this time we impose conditions of the form

$$Z_i \in [a_i, b_i), i \in D$$

The motivation for this problem comes from discrete random fields obtained from quantized Gaussian fields. Given the discrete image, it is desirable to generate a Gaussian field that would result in this image after truncation.

First, consider the case of one variable only. Suppose that we want to generate $Z \sim N(0,1)$ given that $Z \in [a,b)$. The distribution of Z is then given by the density

$$p(z) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{z^2}{2}\right) \frac{I(z \in [a,b))}{\Phi(b) - \Phi(a)}$$

where I is the indicator function and

$$\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{t^2}{2}\right) dt$$

Conceptually, the easiest way to generate Z is the rejection method. Namely, we generate $Z \sim N(0,1)$ and then keep it if it satisfies $Z \in [a,b)$, and discard otherwise. However, the rejection method is highly ineffective. For $[a,b)$ away from the origin, or for a small value of $b-a$, the number of rejections will be too large to be practical. Therefore, for one-dimensional generation we use the Inversion Method (Devroye [20], p. 38):

Lemma 1 *If $U \sim$ Unif$(0,1)$, then $T = \mu + \sigma\Phi^{-1}(\Phi(\frac{a-\mu}{\sigma}) + U[\Phi(\frac{b-\mu}{\sigma}) - \Phi(\frac{a-\mu}{\sigma})])$ has a $N(\mu, \sigma^2)$ distribution, truncated to (a,b), $-\infty \leq a < b \leq \infty$.*

As the dimensionality of $\mathbf{Z}$ grows and the number of conditions rises, the rejection algorithm becomes even less practical. Attempts to modify the Cholesky method to accomodate the conditions on $\mathbf{Z}$ fail. Indeed, it turns out that independent normal ε_i's on which the method is based must now be such that

$$a_i \leq \sum L_{ij}\varepsilon_j < b_i$$

Unfortunately we find generating such ε_i's to be as difficult a problem as the original one. A similar problem makes it impossible to use the circulant embedding method.

An up-to-date summary of conditional generation methods appears in Chilès [11], 1999. Perhaps the most general and straightforward algorithm is sequential simulation. Namely, we sample Z_1 given that it falls between a_1 and b_1, then sample Z_2 given that it falls between a_2 and b_2 *and* given the value of Z_1, and so on up to generating Z_n given that it should fall into $[a_n, b_n)$ and given all the Z_i's, $i < n$. In principle, the method could be applied to any multivariate distribution; the Gaussian distribution is the ideal case in the sense that we know how to calculate the conditional distributions involved (see Ripley [54], p. 99).

For computational reasons (taking into account the size of n), we prefer the Markov chain Monte Carlo (MCMC) technique, as suggested in Freulon and de Fouquet [25]. Contrary to sequential simulation, this method is approximate; however, it is possible to make it utilize the block-circulant structure, and therefore we find it more efficient for programming in this context. We now describe the MCMC method.

3.4.2 Markov Chain Monte Carlo

Introduction

MCMC originated in the statistical physics literature. The purpose of the method was integration in high-dimensional spaces, where the computational requirements of the standard deterministic techniques make them impractical. Suppose that given a random vector $\mathbf{X}$ with the density $p(\mathbf{x})$ supported on $A \in \mathbb{R}^k$ we want to compute the expected value of $f(\mathbf{X})$,

$$E(f(\mathbf{X})) = \int_A f(\mathbf{x})p(\mathbf{x})d\mathbf{x}$$

The idea of the Monte Carlo approach is to replace a statement of the form

> "Given $\varepsilon > 0$, one needs N function evaluations to compute $\int_A f(\mathbf{x})p(\mathbf{x})d\mathbf{x}$ within ε of the true value."

with a probabilistic one:

> "Given $\varepsilon_1 > 0$ and $\varepsilon_2 > 0$, one needs M function evaluations to ensure that with probability at least $1 - \varepsilon_2$ the approximate value of $\int_A f(\mathbf{x})p(\mathbf{x})d\mathbf{x}$ is within ε_1 of the true value."

In a sense, the second statement is weaker than the first one; however, this is a good price to pay for having $M \ll N$.

The idea of simple Monte Carlo is to generate independent identically distributed observations $\mathbf{X}_1, \ldots, \mathbf{X}_n$ from the density p and use $(1/n)\sum f(\mathbf{X}_i)$ to approximate the integral. The problem with this approach is that it is not clear how to generate such $\mathbf{X}_i$'s in cases when the density p is non-standard. Therefore a more sophisticated MCMC method was developed, based on the observation that in order for the approximation to work, $\{\mathbf{X}_i\}$ need not necessarily be independent. The $\{\mathbf{X}_i\}$ can be generated by any process that, loosely speaking, draws samples throughout the support of p in the correct proportions. MCMC does this by constructing a Markov chain having p as its stationary distribution, and gives an algorithm (in fact, a family of algorithms) for generating such Markov chains for any p. We use this part of the MCMC technique without intention to evaluate integrals, but exactly for the sake of approximating distributions, in our case the truncated multivariate normal.

Various versions of MCMC[2]

The most general algorithm for constructing a Markov chain with the given stationary distribution p is the Hastings algorithm (Hastings, [29]). First we select a *proposal* distribution q (from which we know how to sample) and a starting value $\mathbf{X}^{(0)}$. At each step i, the next state $\mathbf{X}^{(i+1)}$ is chosen by first sampling a *candidate* point $\mathbf{Y}$ from $q(\cdot \mid \mathbf{X}^{(i)})$. This candidate point is then *accepted* with probability

$$\alpha(\mathbf{X}^{(i)}, \mathbf{Y}) = \min\left\{1, \frac{p(\mathbf{Y}), q(\mathbf{X}^{(i)} \mid \mathbf{Y})}{p(\mathbf{X}^{(i)}), q(\mathbf{Y} \mid \mathbf{X}^{(i)})}\right\}$$

If the candidate point is accepted, the next state becomes $\mathbf{X}^{(i+1)} = \mathbf{Y}$. If the candidate is rejected, the chain does not move, i.e. $\mathbf{X}^{(i+1)} = \mathbf{X}^{(i)}$. Under general regularity conditions it can be proved that the resulting Markov chain will have the stationary distribution p regardless of the choice of q. However, q affects the rate of convergence and the *mixing* – the speed with which the chain moves around in the support of p after having 'converged'. The tradeoff is between how close the shape of the proposal distribution q to that of p is, and how easy it is to sample from it.

[2]Our main source on MCMC methodology is Gilks et al. [27].

Different MCMC algorithms are categorized based on the properties of the proposal distribution. If q is symmetric, that is, if $q(\mathbf{Y} \mid \mathbf{X}) = q(\mathbf{X} \mid \mathbf{Y})$, the MCMC with this q is called the *Metropolis algorithm*. Another example would be the *independence sampler* (Tierney, [59]), a MCMC algorithm whose proposal $q(\mathbf{Y} \mid \mathbf{X}) = q(\mathbf{Y})$ does not depend on $\mathbf{X}$.

Another variation of the general algorithm is the *single-component MCMC* – the original framework proposed by Metropolis et al. [46]. Instead of updating the whole $\mathbf{X}$ *en bloc*, it is sometimes more convenient and efficient to divide it into h components $\mathbf{X}_1, \dots, \mathbf{X}_h$ of possibly different dimensions. Transition from state i of the chain to state $i+1$ consists in this case of updating the h components $\mathbf{X}_k^{(i)}$ of $\mathbf{X}^{(i)}$ one at a time, $k = 1, \dots, h$. For each component its own proposal distribution q_k is used, and q_k may depend on the current state of this and all the other components. Namely, the candidate $\mathbf{Y}_k$ is sampled from $q_k(\cdot \mid \mathbf{X}_1^{(i+1)}, \dots, \mathbf{X}_{k-1}^{(i+1)}, \mathbf{X}_k^{(i)}, \dots, \mathbf{X}_h^{(i)})$. The candidate is accepted and replaces $\mathbf{X}_k^{(i)}$ with probability

$$\alpha = \min\left\{1, \frac{p\left(\mathbf{Y}_k \mid \mathbf{X}_1^{(i+1)}, \dots, \mathbf{X}_{k-1}^{(i+1)}, \mathbf{X}_k^{(i)}, \dots, \mathbf{X}_h^{(i)}\right)}{p\left(\mathbf{X}_k^{(i)} \mid \mathbf{X}_1^{(i+1)}, \dots, \mathbf{X}_{k-1}^{(i+1)}, \mathbf{X}_k^{(i)}, \dots, \mathbf{X}_h^{(i)}\right)} \times \frac{q_k\left(\mathbf{X}_k^{(i)} \mid \mathbf{X}_1^{(i+1)}, \dots, \mathbf{X}_{k-1}^{(i+1)}, \mathbf{Y}_k, \mathbf{X}_{k+1}^{(i)}, \dots, \mathbf{X}_h^{(i)}\right)}{q_k\left(\mathbf{Y}_k \mid \mathbf{X}_1^{(i+1)}, \dots, \mathbf{X}_{k-1}^{(i+1)}, \mathbf{X}_k^{(i)}, \mathbf{X}_{k+1}^{(i)}, \dots, \mathbf{X}_h^{(i)}\right)}\right\} \tag{3.3}$$

If $\mathbf{Y}_k$ is not accepted, the k-th component remains the same: $\mathbf{X}_k^{(i+1)} = \mathbf{X}_k^{(i)}$.

Now we are ready to define the *Gibbs sampler*, an MCMC technique that we use for generation of truncated multivariate normal vectors.

Gibbs Sampler[3]

For an n-variate distribution p, the *full conditionals* are defined as the univariate conditional distributions derived from p when all the variables but one are fixed. Thus, for $\mathbf{X} \sim p$, the k-th full conditional is the conditional distribution of the coordinate X_k given all the other coordinates:

$$X_k \sim p(\cdot \mid X_1, \ldots, X_{k-1}, X_{k+1}, \ldots, X_n), k = 1, \ldots, n$$

The Gibbs sampler is a single-component MCMC algorithm, in which $\mathbf{X}$ is divided into univariate components X_k, and all the proposal distributions q_k are full conditionals of $\mathbf{X}$. In this case the expression (3.3) for α reduces to

$$\alpha = \min\{1, 1\} \equiv 1$$

so that the proposals in the Gibbs sampler are always accepted. This simplifies the algorithm, but the Gibbs sampler can be used only if it is possible to sample from the full conditionals.

Example: Suppose that we want to sample $(X, Y, Z) \sim p(x, y, z)$ using the Gibbs sampler. The procedure is as follows:

- Choose $X^{(0)}, Y^{(0)}, Z^{(0)}$;
- Sample $X^{(1)} \sim p(x \mid y = Y^{(0)}, z = Z^{(0)})$;

  Sample $Y^{(1)} \sim p(y \mid x = X^{(1)}, z = Z^{(0)})$;

  Sample $Z^{(1)} \sim p(z \mid x = X^{(1)}, y = Y^{(1)})$;
- $\ldots$ (do the same replacing 0 by $i - 1$ and 1 by i)
- For large n, the distribution of $(X^{(n)}, Y^{(n)}, Z^{(n)})$ is approximately p.

[3]The Gibbs sampler is described in detail in Casella and George [10].

3.4.3 Using Gibbs Sampler to Generate Truncated Gaussian Fields

We now return to the problem of generating a sample from a zero-mean Gaussian field $\mathbf{Z}$ with the given covariance function r over an $n_1 \times n_2$ grid $\mathbf{S} = \{\mathbf{s}_{kj}\}$, subject to the conditions

$$a_{kj} \leq \mathbf{Z}(\mathbf{s}_{kj}) < b_{kj} \tag{3.4}$$
$$-\infty \leq a_{kj}, b_{kj} \leq \infty.$$

That is, for each location we specify its own interval for a field value.

In the same way as in Section 3.3.2, we combine $\mathbf{Z}(\mathbf{s}_{kj})$ in a vector $\mathbf{Z}$ of dimension $n_1 n_2$. The covariance function r and the grid $\mathbf{S}$ together yield a covariance matrix $\mathbf{T}$. For the same computational reasons as in Section 3.3.2, instead of generating a truncated $\mathbf{Z} \sim N(\mathbf{0}, \mathbf{T})$, we generate a $4n_1n_2$ dimensional vector $\mathbf{W} \sim N(\mathbf{0}, \mathbf{C})$, where $\mathbf{C}$ is the block circulant matrix with circulant blocks that embeds $\mathbf{T}$ and then extract the components of $\mathbf{W}$ that correspond to $\mathbf{Z}$. In the process of generation, these components are subject to (3.4), while the others are left unrestricted.

To implement the Gibbs sampler for $\mathbf{W}$ we must identify its full conditionals. Let

$$I_k(w) = \begin{cases} 1, & \text{if } W_k \text{ does not correspond to any of the } Z_j\text{'s (i.e., no restriction on } W_k\text{)}, \\ I(a_j \leq w < b_j), & \text{otherwise.} \end{cases}$$

Then, for $(w_1, w_2, \ldots, w_{k-1}, w_{k+1}, \ldots, w_n)$ satisfying the conditions (3.4) on

$\mathbf{Z}$, the full conditional distribution of W_k is given by

$$\begin{aligned}
&p_{W_k}(w|w_1, w_2, \ldots, w_{k-1}, w_{k+1}, \ldots, w_n) \\
&\quad \propto p_{\mathbf{W}}(w_1, w_2, \ldots, w_{k-1}, w, w_{k+1}, \ldots, w_n) I_k(w) \\
&\quad = \frac{1}{(2\pi)^{n/2}\sqrt{|C|}} \exp\left(-\frac{1}{2}(w_1, w_2, \ldots, w_{k-1}, w, w_{k+1}, \ldots, w_n)\mathbf{C}^{-1} \right. \\
&\qquad\qquad\qquad \left. \times (w_1, w_2, \ldots, w_{k-1}, w, w_{k+1}, \ldots, w_n)^T \right) I_k(w) \\
&\quad \propto \exp\left(-\frac{d_{kk}w^2 + 2\sum_{j\neq k} d_{jk}w_j w}{2} \right) I_k(w) \\
&\quad \propto \exp\left(-\frac{\left(w + \sum_{j\neq k} d_{jk}w_j/d_{kk}\right)^2}{2/d_{kk}} \right) I_k(w),
\end{aligned}$$

where d_{jk} are the elements of $\mathbf{C}^{-1}$. This shows that

$$W_k|W_{j,\, j\neq k} \sim N\left(-\sum_{j\neq k} d_{jk}w_j/d_{kk}, 1/d_{kk} \right)$$

restricted to $[a_j, b_j)$ if W_k corresponds to certain Z_j, and unrestricted otherwise.

Recall (Section 2.7.3) that $\mathbf{C}^{-1}$ is block circulant with circulant blocks, as is $\mathbf{C}$. Therefore its k-th column is found easily from its first column, which, in turn, is found from the first column of $\mathbf{C}$, so that $\sum_{j\neq k} d_{jk}w_j$ is available. Moreover, because of the structure of $\mathbf{C}^{-1}$, the values along the main diagonal are constant, so that each d_{kk} is equal to d_{11} which we know once we find the first column of $\mathbf{C}^{-1}$.

Thus the problem of generating a truncated Gaussian field is reduced to sampling

$$W \sim N(\mu, \sigma^2) \text{ given } W \in [a, b)$$

the method for which is given in Lemma 1 of Section 3.4.1.

3.5 Simulations

To test the algorithms presented in this chapter, we simulated Gaussian fields conditionally and unconditionally. In each case, 250 zero-mean Gaussian fields $\mathbf{Z}^{(1)}, \ldots, \mathbf{Z}^{(250)}$ were generated on a 32×32 grid with the isotropic exponential covariance function

$$r(s) = e^{\theta s}, \quad \theta = -0.3,$$

where s is the distance between points. In the conditional case, the points in the even-numbered columns of the grid were restricted to the interval $(0, 1)$.

The C program was compiled with `gcc` under Digital Unix on DEC Alpha. The FFT was performed by the FFTW library (see Section 2.8). Uniform (0,1) random numbers where generated with the standard `drand48` function from the `stdlib` library. From those, normal variables where generated by the following algorithm:

- Generate $U_1, U_2 \sim \text{Unif}(0, 1)$.
- Define $V_1 = 2U_1 - 1, \quad V_2 = 2U_2 - 1$.
- If $r = V_1^2 + V_2^2 > 1$, start over.
  Otherwise $X_i = V_i\sqrt{-2\log r/r}, i = 1, 2$ are independent N(0, 1) variables ([9], p.194).

Inverse normal distribution function required for the Inversion Method (Lemma 1) used for generating truncated univariate normals was computed with the `z.c` module written by G. Perlman [50].

For each field, three sequences of 12 points each were recorded as shown on Figure 3.2.

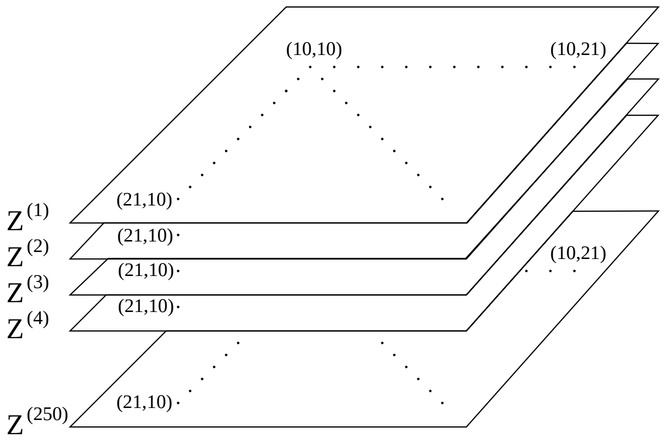

Figure 3.2: Field values at these points where recorded in 250 simulations.

Figures 3.3 and 3.5 show histograms of the 250 realizations of field values at the first six points of each sequence, for conditional and unconditional simulations respectively. For unconditional simulation, the sample covariance function estimated from the recorded observations was plotted against the theoretical covariance function r (Figure 3.4).

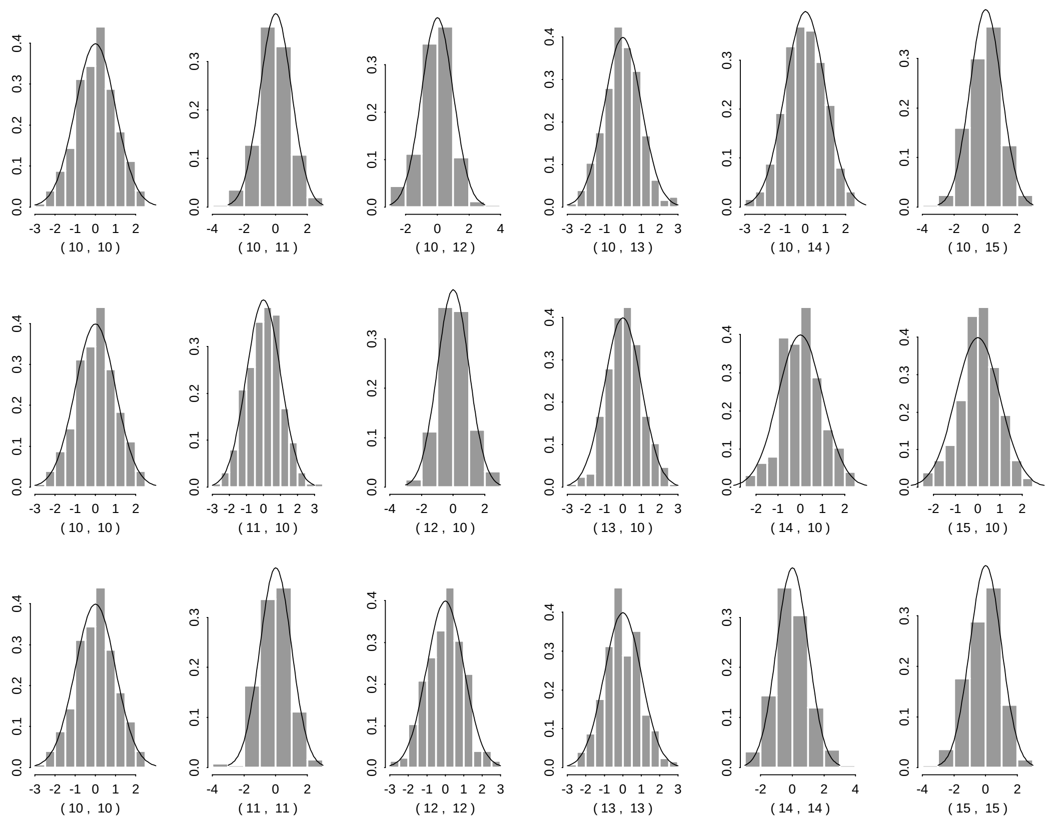

Figure 3.3: Unconditional generation of Gaussian field. Histograms of 250 realizations of the first six points from each of the three sequences (see Figure 3.2) are presented, with superimposed $N(0, 1)$ density function. Each location of the field appears close to being $N(0, 1)$ distributed.

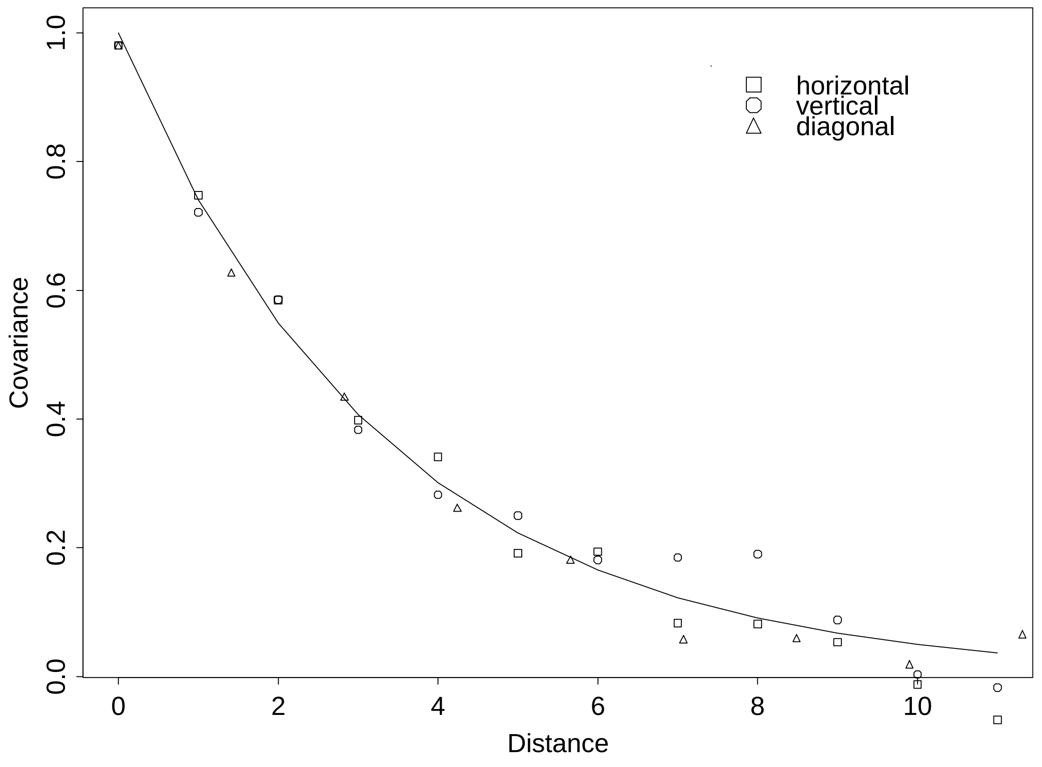

Figure 3.4: Unconditional generation of Gaussian field. Squares, circles, and triangles represent sample covariances of $Z_{(10,10)}$ with the points in the horizontal, vertical, and diagonal sequences respectively (see Figure 3.2). Hence, the x coordinates of circles and squares are integers, while those of the triangles are multiples of $\sqrt{2}$. The solid line is the plot of the theoretical covariance function $r(s) = \exp(-0.3s)$.

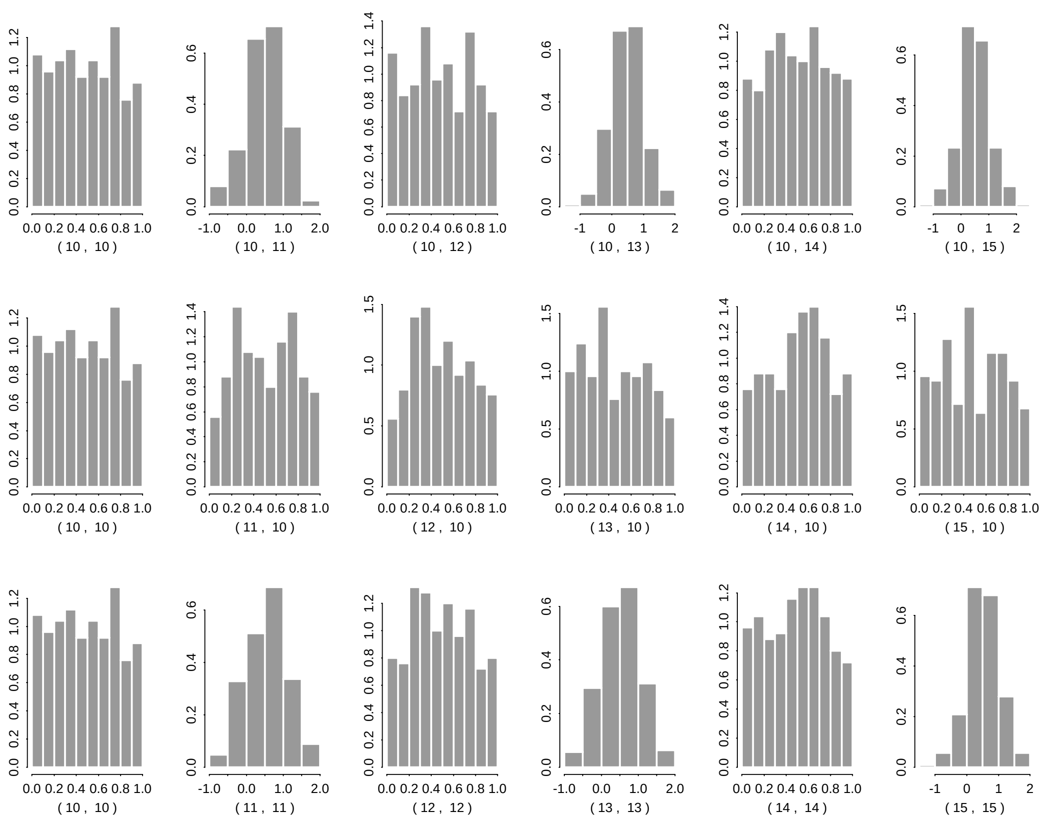

Figure 3.5: Conditional generation of a Gaussian field given that all the points in the even columns of the grid fall into $(0, 1)$. Histograms of 250 realizations of the first six points from each of the three sequences (see Figure 3.2) are presented. The range of the restricted components is correct; unrestricted components have bell-shaped distributions with positive means and variances smaller than those in Figure 3.3.

Chapter 4

DISCRETE IMAGES AS CLIPPED GAUSSIAN FIELDS

4.1 Introduction

In this chapter we discuss an approach to modeling and estimation in spatial processes whose realizations can be represented by color maps with a small number of colors (in particular, binary images). Examples where this sort of situation occurs are numerous: a geologic formation composed of several rock types, a part of an ocean surface made up of ice and water, a contaminated geographic region with subregions defined as locations where the contaminant concentration surpasses certain safety levels, or a quantized rain rate snapshot. In an analogy to discrete and continuous random fields, we call such images 'discrete', as opposed to 'continuous color' or 'gray-scale' images. Strictly speaking, any image stored in a computer is discrete (e.g., 256 shades of gray). However, the distinction is the same as in the case of regular discrete and random variables.

We model processes resulting in discrete images by clipping stationary zero-mean Gaussian fields at several levels, called thresholds. By choosing different covariance functions and thresholds, we are able to produce a large variety of textures (See Figures 4.1 - 4.4).

Going the other way, suppose a scientist gives us a single discrete image. We can estimate the clipping levels by comparing the observed areal fractions to the quantiles of the normal distribution, make an assumption about the parametric family of covariance functions in the underlying Gaussian field, and estimate the parameters. Now we can obtain new images (supposedly having the same properties as the original one) by generating Gaussian fields with the estimated parameters, and clipping them at the estimated thresholds. Then we can decide whether the model is satisfactory asking the scientist how 'real' the generated images look.

Generating similar images will also help to study the algorithms that are used on the images. The idea is reminiscent of parametric bootstrapping in the sense that new samples (discrete images) are generated using the estimated parameters from a given sample (original discrete image) for estimating the variability and precision of estimators.

In practice some discrete images are actually produced (either by choice or by the characteristics of the recording device) by thresholding an underlying process, as in the contaminant and rain rate examples. However, even if there is no clear physical process producing image, thresholding can still be used in many situations to model discrete spatial data.

4.2 A TRMM Application

The Tropical Rainfall Measuring Mission (TRMM), was launched on November 27, 1997, by placing in a low earth orbit of 350 km a satellite that is expected to collect rainfall and related data for at least three years. It is a joint mission of the National Aeronautics and Space Administration (NASA) and the National Space Development Agency (NASDA) of Japan whose goal is to study the effect of tropical rainfall and the associated energy release on the global atmospheric circulation. Regarding rainfall, TRMM produces instantaneous rain rate snapshots over large areas, obtained from an array of spaceborne instruments including a precipitation radar (PR) and the TRMM microwave imager (TMI). The snapshots are produced along a swath of width 750 km (TMI) and 220 km (PR). The problem is to get the monthly mean rain rate over $5^\circ \times 5^\circ$ boxes. See

`http://trmm.gsfc.nasa.gov/trmm_office/index.html`

for more details. Now, the TRMM satellite visits random sub-areas of any given $5^\circ \times 5^\circ$ box only a few times, perhaps once a day or 30 times during a month. Also, to render the data more reliable, they are categorized or quantized as explained by Kedem, Pfeiffer and Short [35]. This means that the data may be viewed as a collection of discrete spatial images. Consider the area average of one such image. To estimate the *variability* of the area average, we can use parametric bootstrapping as explained above. Namely, we generate from a given discrete image many discrete images all having the same statistical properties for the purpose of estimating the variance of the area average of rain rate obtained from an original discrete image. This can help in the estimation of the space-time monthly mean rain rate of $5^\circ \times 5^\circ$ boxes.

4.3 The model

We view a k-color $n_1 \times n_2$ image $\mathbf{X}$ as a realization of a discrete random field taking k different values $0, 1, \ldots, k-1$ over a grid $\mathbf{S} = \{s_{ij}\}$. The field $\mathbf{X}$ is modeled in terms of an unobserved stationary Gaussian field $\mathbf{Z}$ with mean zero, variance one, and covariance function r depending on the vector parameter $\boldsymbol{\theta}$. Then $\mathbf{Z}$ defines $\mathbf{X}$ in the following way. For a vector of thresholds $\mathbf{c} = (c_0, c_1, \ldots, c_k)$, the field $\mathbf{X}$ is a quantization of $\mathbf{Z}$ at levels $\mathbf{c}$. That is,

$$X_i = j \text{ whenever } c_j \leq Z_i < c_{j+1}, \;\; j = 0, \ldots, k-1,$$

where $Z_i \sim N(0,1)$, $\mathrm{Cov}(Z_i, Z_j) = r(s_i, s_j, \boldsymbol{\theta})$, $c_0 = -\infty$, and $c_k = \infty$.

These are some of the commonly used covariance function families (l is the distance between points where the covariance is computed).

Exponential correlation:

$$r_{\boldsymbol{\theta}}(l) = \theta_1^{l^{\theta_2}},$$

where $\theta_1 \in (0,1)$ and $\theta_2 \in (0,2]$.

Matérn correlation:

$$r_{\boldsymbol{\theta}}(l) = \begin{cases} \dfrac{1}{2^{\theta_2 - 1}\Gamma(\theta_2)} \left(\dfrac{l}{\theta_1}\right)^{\theta_2} \mathcal{K}_{\theta_2}\left(\dfrac{l}{\theta_1}\right), & l \neq 0 \\ 1, & l = 0, \end{cases}$$

where $\theta_1 > 0$, $\theta_2 > 0$ and $\mathcal{K}_{\theta_2}$ is a modified Bessel function of the third kind of order θ_2.

Rational quadratic correlation:

$$r_{\boldsymbol{\theta}}(l) = \left(1 + \frac{l^2}{\theta_1^2}\right)^{-\theta_2},$$

where $\theta_1 > 0$ and $\theta_2 > 0$.

Spherical correlation:

$$r_\theta(l) = \begin{cases} 1 - \frac{3}{2}\left(\frac{l}{\theta}\right) + \frac{1}{2}\left(\frac{l}{\theta}\right)^3, & l \leq \theta \\ 0, & \text{otherwise} \end{cases}$$

where $\theta > 0$.

Figures 4.1 – 4.4 illustrate various three-color patterns obtained by clipping Gaussian fields with these covariance functions at levels that divide the normal distribution range 'equally': $c_1 = \Phi^{-1}(1/3) \approx -0.43, \quad c_2 = \Phi^{-1}(2/3) \approx 0.43$.

Figures 4.5 – 4.8 provide examples of different realizations of discrete fields with the same $\boldsymbol{\theta}$. More images can be generated online at

`http://www.math.umd.edu/~bak/gaussian/generate.cgi`

The modified Bessel function for Matérn correlation in Figures 4.3, 4.5, and 4.8 was computed with the `rkbesl` routine from the SPECFUN FORTRAN package ([12]), translated into C by D. Bindel [5].

4.4 Estimation

Suppose we are given a discrete image and want to estimate the model parameters $\mathbf{c}$ and $\boldsymbol{\theta}$. First of all we estimate the thresholds $\mathbf{c}$ if they are unknown. There is no unique or best way of doing that; our choice is to compare the observed

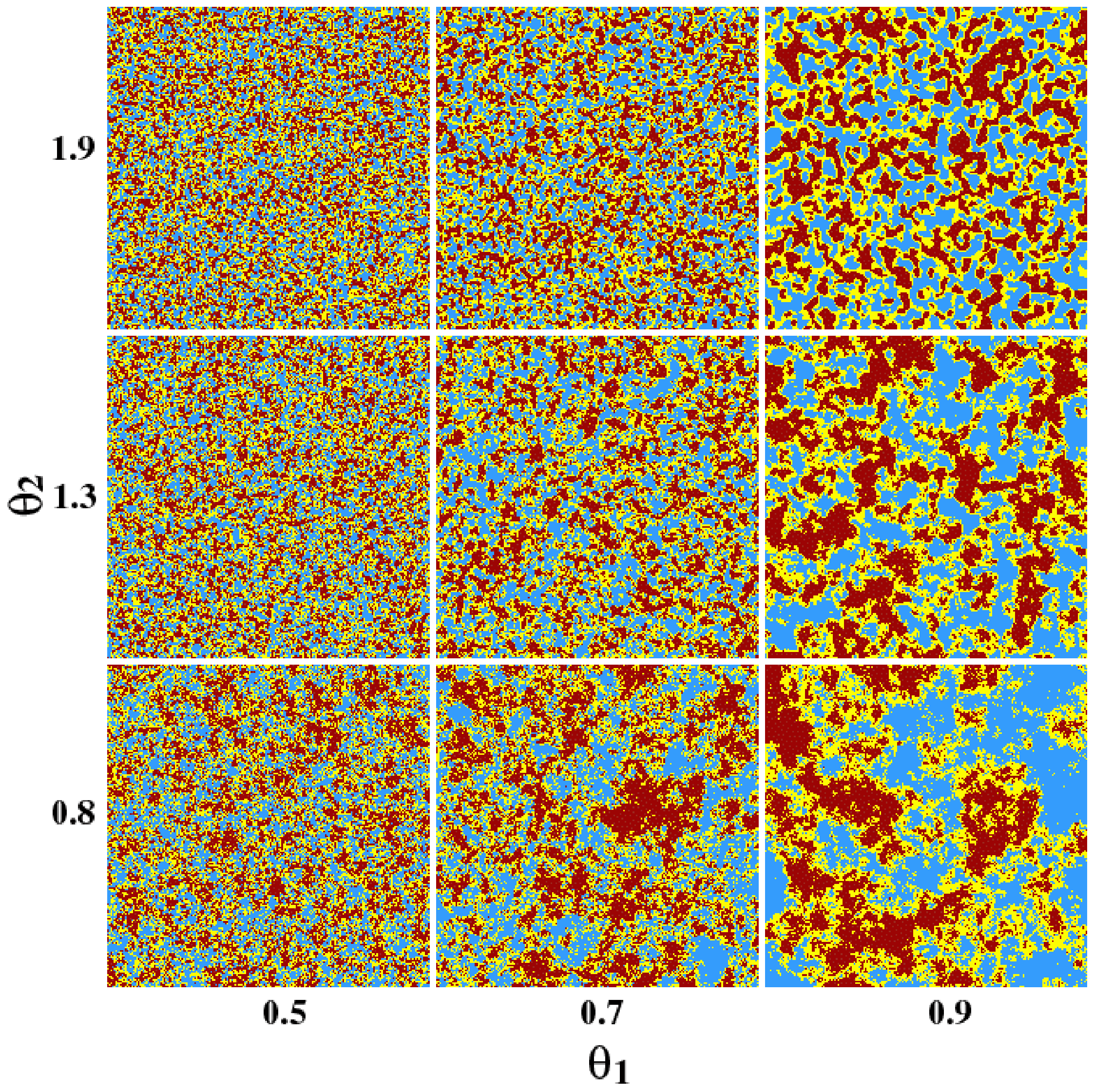

Figure 4.1: Three color patterns obtained by clipping Gaussian fields with exponential covariance functions at levels {-0.43, 0.43}. The sizes of the connected regions increase in θ_1 and decreases in θ_2.

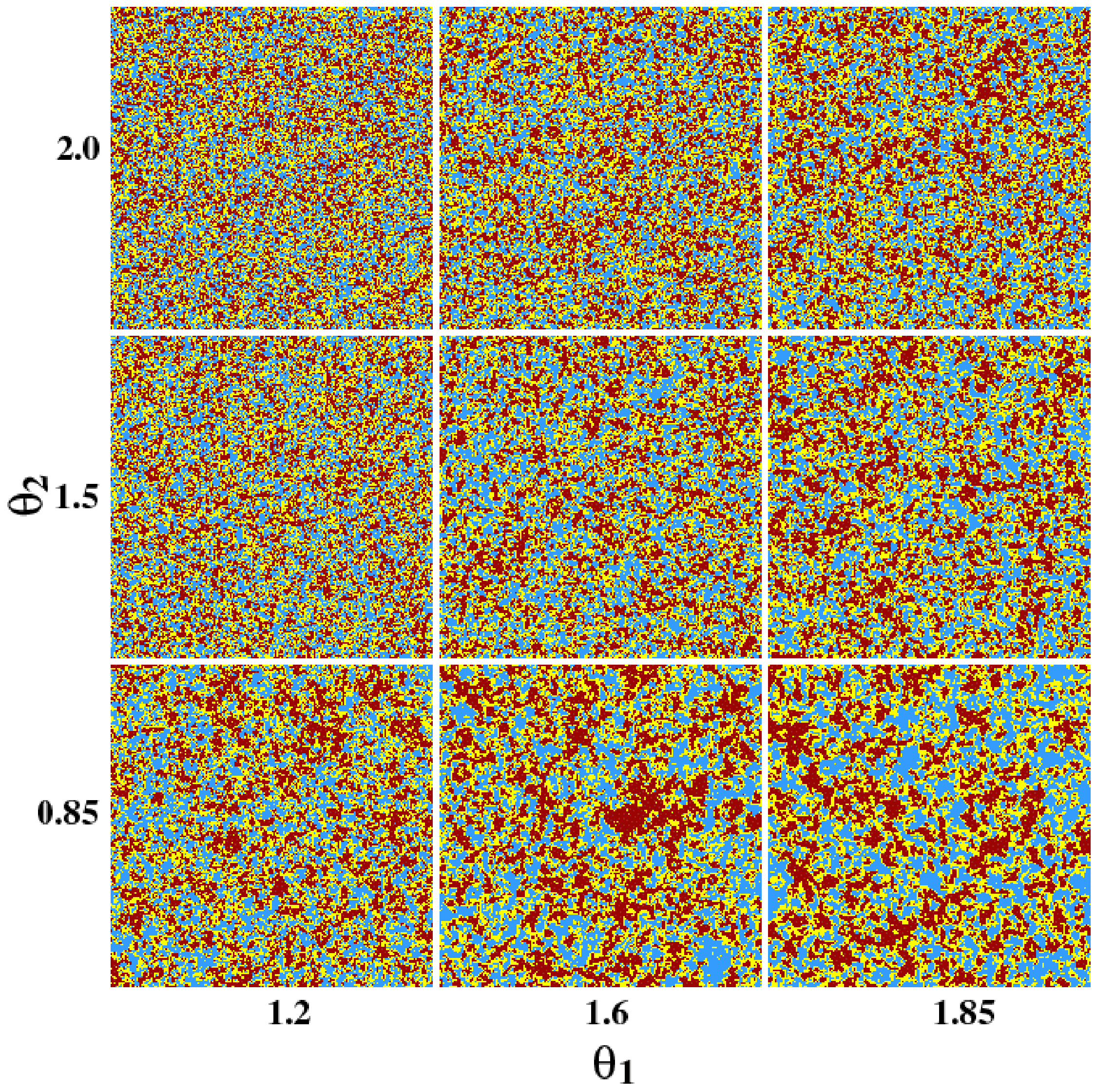

Figure 4.2: Three color patterns obtained by clipping Gaussian fields with rational quadratic covariance functions at levels {-0.43, 0.43}. The sizes of the connected regions increase in θ_1 and decreases in θ_2.

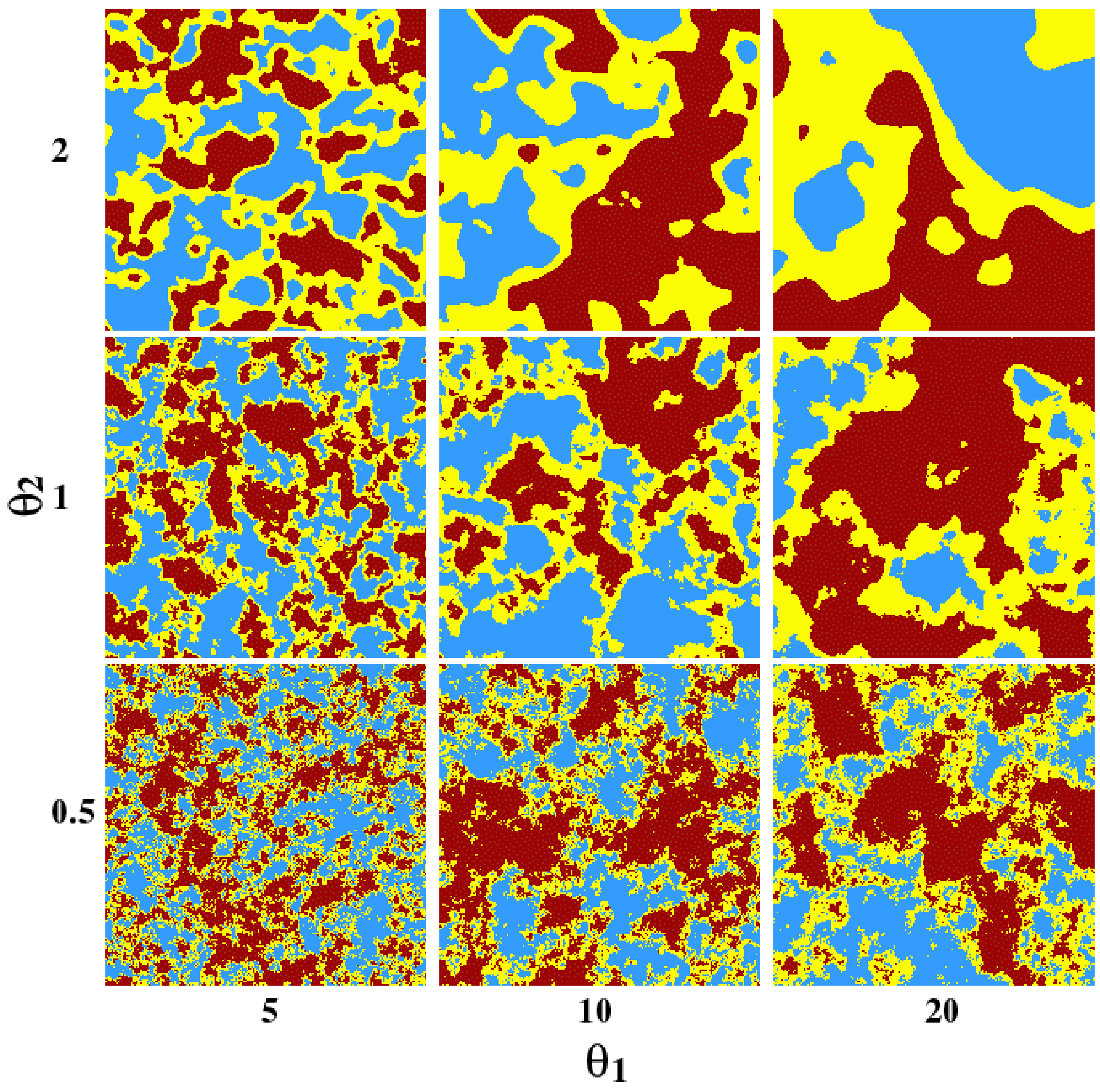

Figure 4.3: Three color patterns obtained by clipping Gaussian fields with Matérn covariance functions at levels {-0.43, 0.43}. The sizes of the connected regions increase in θ_1 and θ_2.

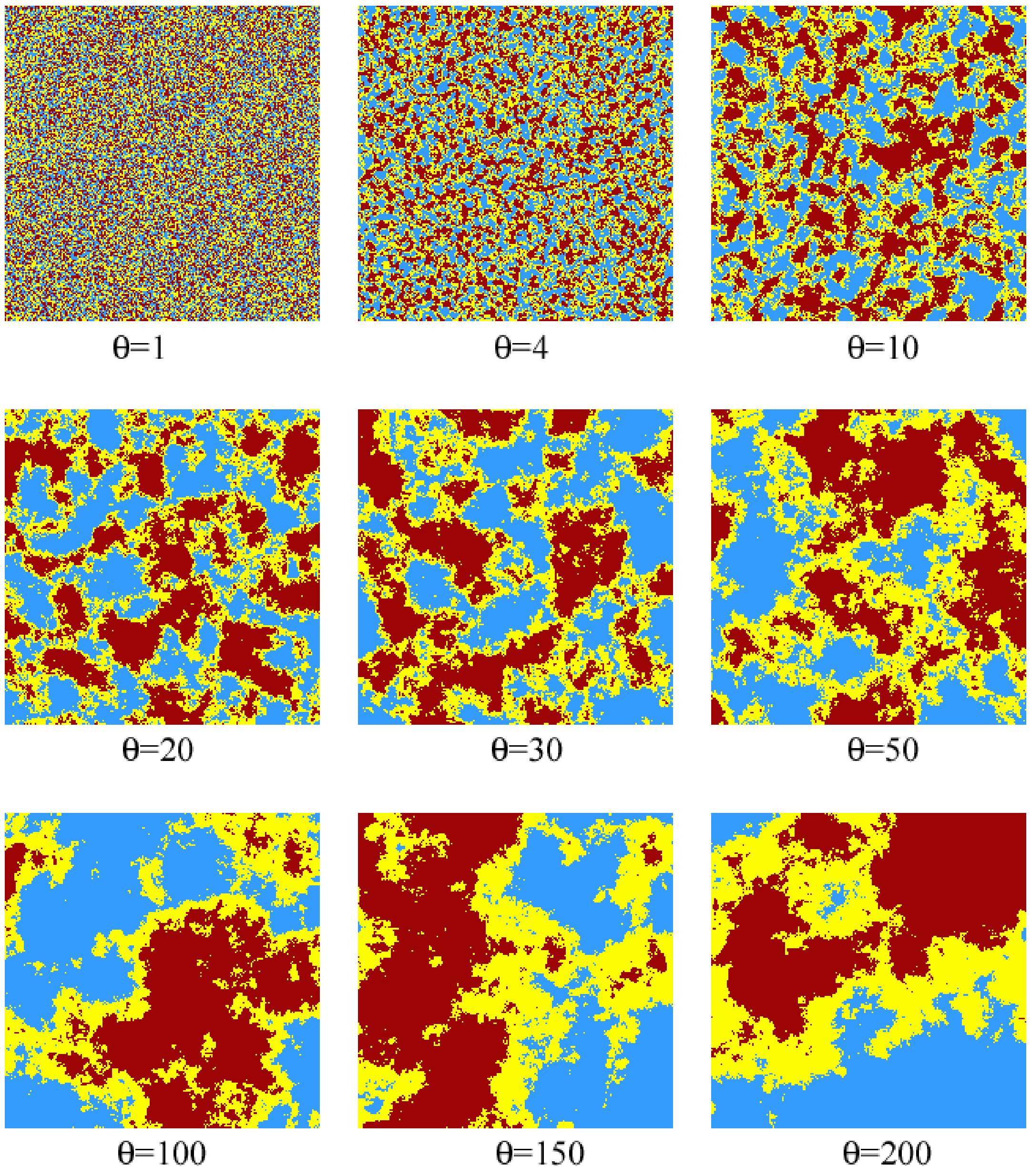

Figure 4.4: Three color patterns obtained by clipping Gaussian fields with spherical covariance functions at levels {-0.43, 0.43}. The sizes of the connected regions increase in θ.

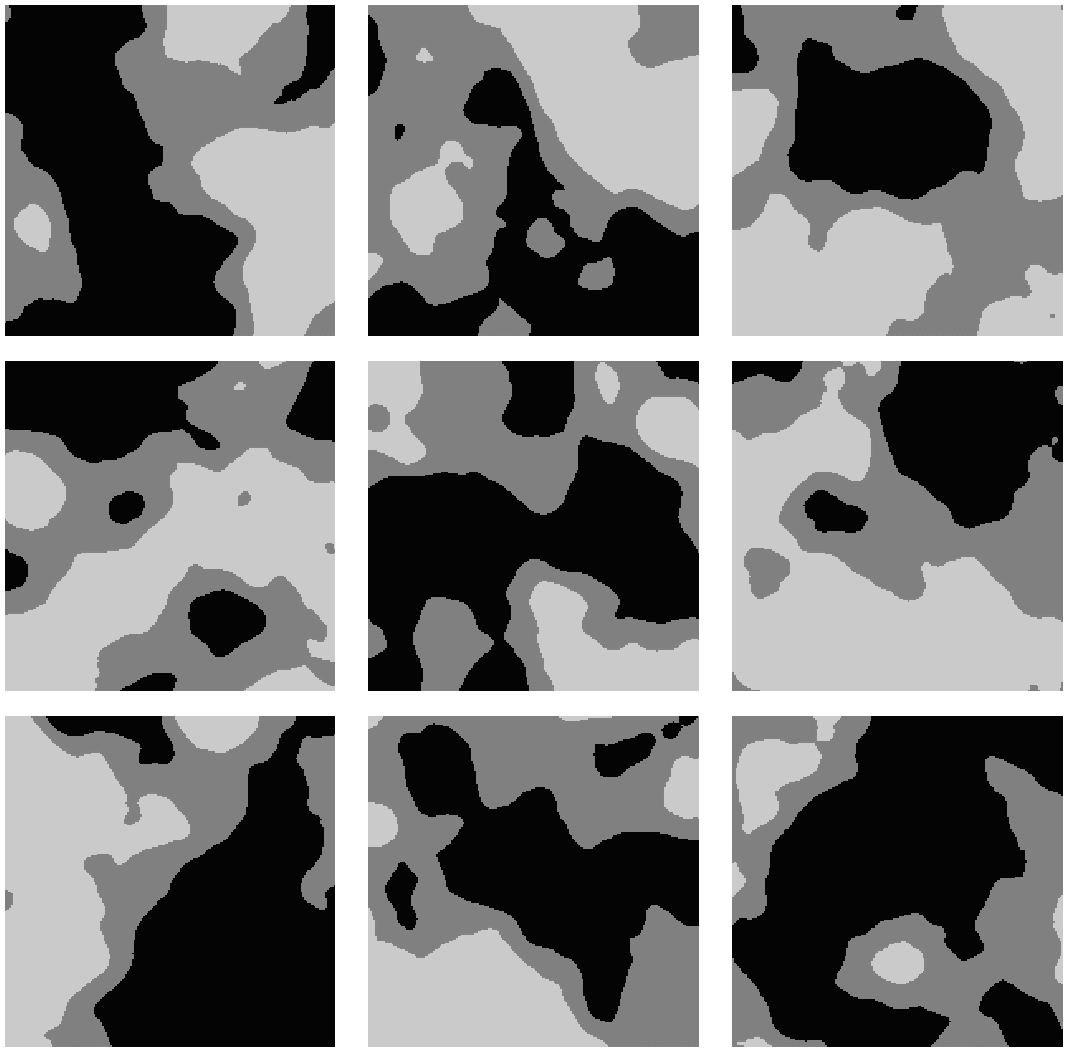

Figure 4.5: Nine realizations of the tree color field obtained by clipping a Gaussian field with Matérn covariance function, $\boldsymbol{\theta} = (20, 2)$, at levels {-0.43, 0.43}.

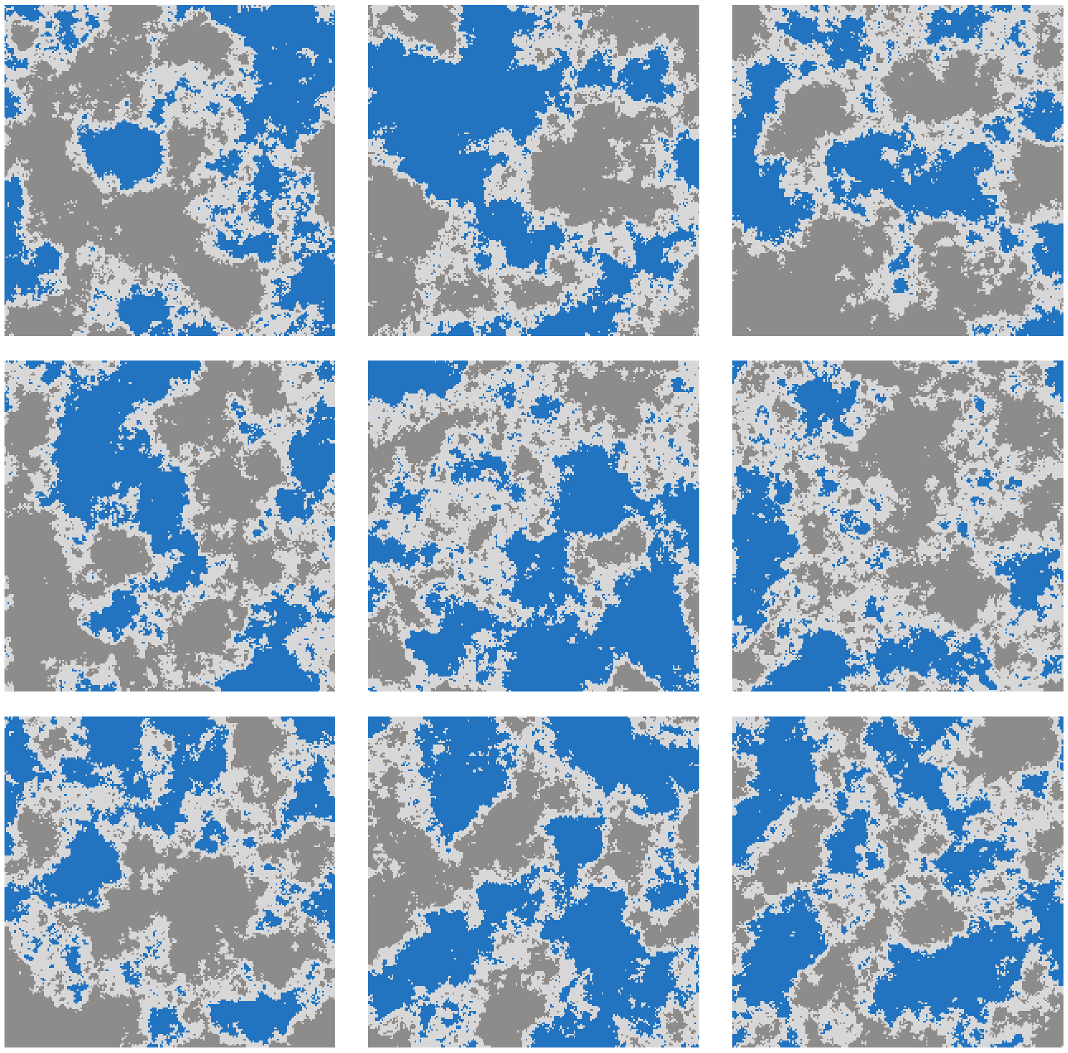

Figure 4.6: Nine realizations of the three color field obtained by clipping a Gaussian field with spherical covariance function, $\theta = 50$, at levels {-0.43, 0.43}.

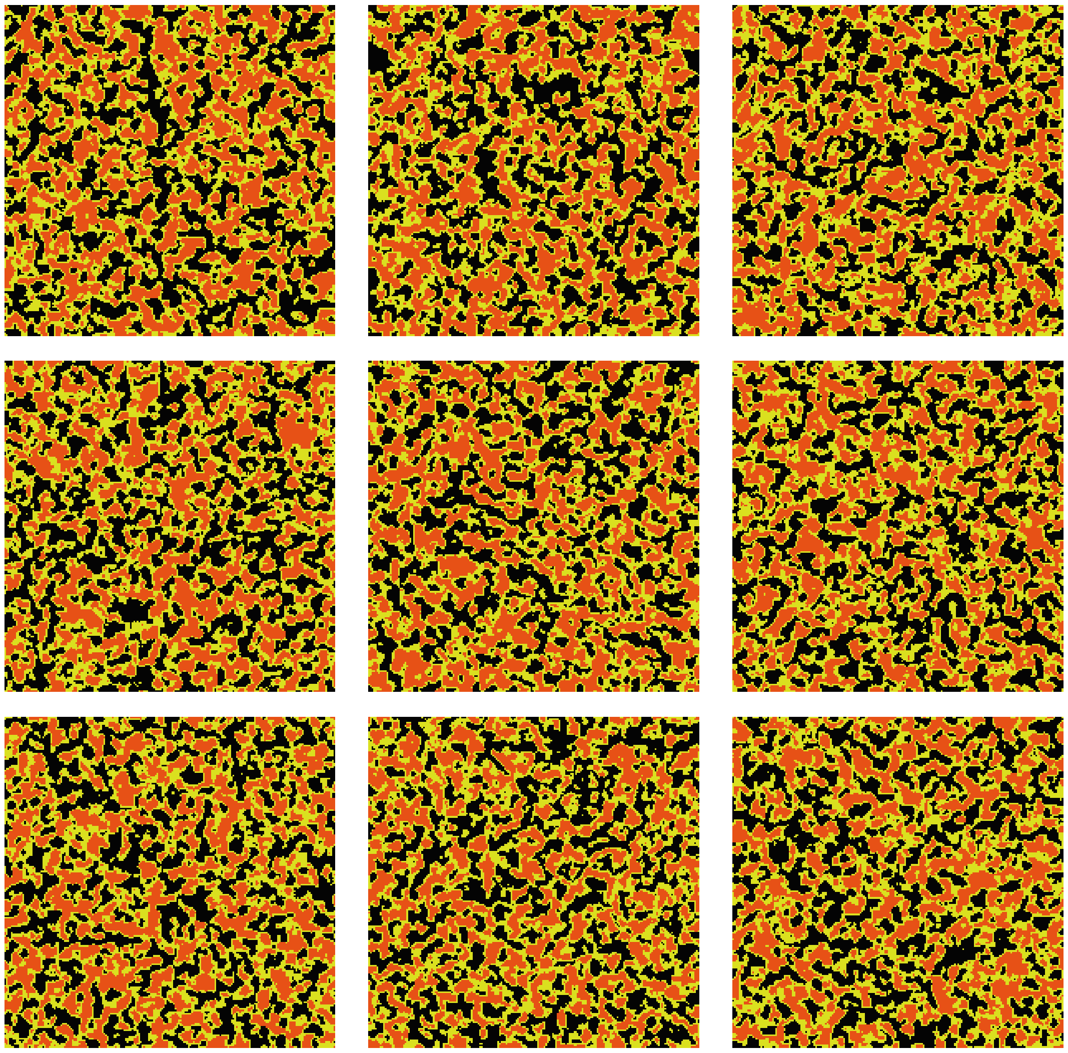

Figure 4.7: Nine realizations of the three color field obtained by clipping a Gaussian field with exponential covariance function, $\boldsymbol{\theta} = (0.9, 1.9)$, at levels {-0.43, 0.43}.

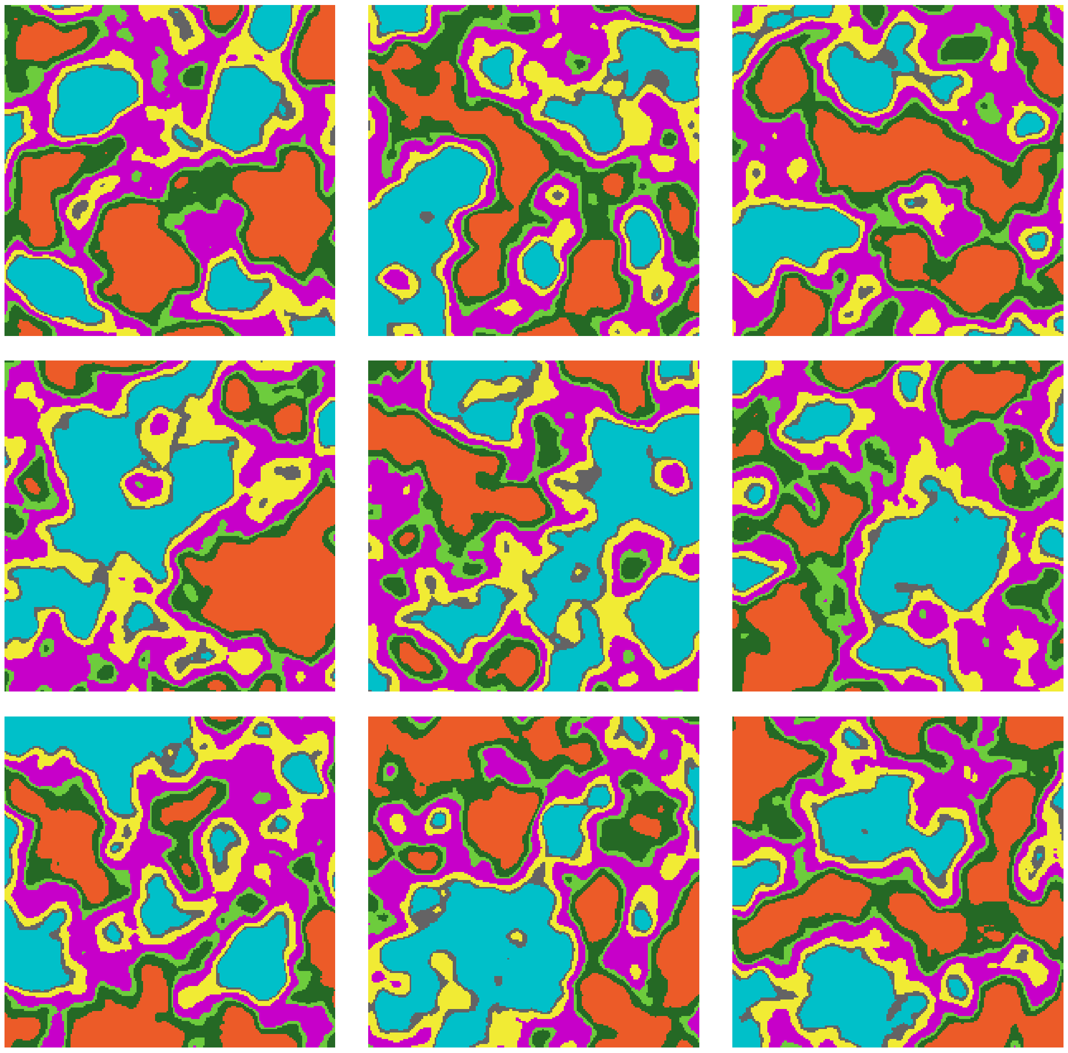

Figure 4.8: Nine realizations of the seven color field obtained by clipping a Gaussian field with Matérn covariance function, $\boldsymbol{\theta} = (10, 2)$.

fraction of each color in the image to quantiles of the normal distribution:

$$c_i = \Phi^{-1}\left(\frac{\#\{X_j < i\}}{n_1 n_2}\right), \quad i = 1, \ldots, k-1.$$

Comparing the given image visually to the known patterns, or based on additional scientific or physical considerations, we select a parametric family for the covariance function r. Now the vector parameter $\boldsymbol{\theta}$ has to be estimated.

A natural approach to estimation of $\boldsymbol{\theta}$ is the EM (expectation-maximization) algorithm (Dempster et al., [19]) with the complete data $\mathbf{Z}$ (unobserved) and incomplete data $\mathbf{X}$ (observed). The EM algorithm starts with an initial estimate $\boldsymbol{\theta}^{(0)}$ and updates it iteratively by the rule

$$\boldsymbol{\theta}^{(n+1)} = \arg\ \max_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}^{(n)}}\big(\log L(\mathbf{Z};\boldsymbol{\theta} \mid \mathbf{X})\big)$$

(note that $\mathbf{X}$ is a function of $\mathbf{Z}$). It can be shown that at each step of the EM algorithm the likelihood of $\boldsymbol{\theta}$ under the observed data does not decrease. Therefore it is hoped that the process would converge to the value $\boldsymbol{\theta}^*$ that maximizes the likelihood of $\boldsymbol{\theta}$ under the observed data. In reality, however, the algorithm only converges to some stationary point of the likelihood, which could be a local maximum or a saddle point. Convergence theory for the EM algorithm is given in Wu [64]. Our approach to the convergence problem is discussed in Section 4.5.

Since it is not clear how to compute the conditional expectation involved, we use a Monte-Carlo EM (MCEM, see McLachlan [44], p. 214) version, in which the following approximation is used:

$$E_{\boldsymbol{\theta}^{(n)}}\big(\log L(\mathbf{Z};\boldsymbol{\theta} \mid \mathbf{X})\big) \approx \frac{1}{s}\sum_{i=1}^{s} \log L(\mathbf{Z}^{(i)};\boldsymbol{\theta})$$

Here $\{\mathbf{Z}^{(i)}\}$ is an ergodic sequence of realizations of the unobserved data, given $\mathbf{X}$ and with the parameter vector fixed at $\boldsymbol{\theta}^{(n)}$. Even the version with $s = 1$ can be used with success, in which case the method is called Stochastic EM (SEM, see Diebolt and Ip, [21]).

Implementation of the Monte-Carlo EM algorithm presents two difficulties. First, we have to generate $\mathbf{Z}$ given $(\mathbf{X}, \boldsymbol{\theta}^{(n)})$, and second, we have to evaluate the likelihood of various $\boldsymbol{\theta}$'s under this $\mathbf{Z}$ fast enough to make the maximization possible.

The problem of generating $\mathbf{Z}$ is discussed in Section 3.4. As shown there, instead of $\mathbf{Z}$, one should generate a larger vector $\mathbf{W}$ having a block circulant covariance matrix $\mathbf{C}$ with circulant blocks and then extract $\mathbf{Z}$ from $\mathbf{W}$. However, the matrix $\mathbf{C}$ is also highly suitable for computations, in particular, for likelihood evaluation, contrary to the block Toeplitz covariance matrix of $\mathbf{Z}$. Indeed, since

$$L(\boldsymbol{\theta} \mid \mathbf{w}) = (2\pi)^{-n/2} |\mathbf{C}(\boldsymbol{\theta})|^{-1/2} \exp\left(-\mathbf{w}^T \mathbf{C}(\boldsymbol{\theta})^{-1} \mathbf{w}/2\right)$$

where $n = 4n_1n_2$, the log likelihood of $\boldsymbol{\theta}$ under $\mathbf{W}$ is given up to an additive constant by

$$\log L(\boldsymbol{\theta} \mid \mathbf{w}) = -0.5 \left(\log |\mathbf{C}(\boldsymbol{\theta})| + \mathbf{w}^T \mathbf{C}(\boldsymbol{\theta})^{-1} \mathbf{w}\right)$$

Here both $|\mathbf{C}(\boldsymbol{\theta})|$ and $\mathbf{w}^T \mathbf{C}(\boldsymbol{\theta})^{-1} \mathbf{w}$ can be computed by FFT using the technique of Section 2.7, which turns out to be fast enough for maximization. Therefore, we abandon $\mathbf{Z}$ altogether, and treat $\mathbf{W}$ as the unobserved data throughout the EM algorithm setup. As Nott and Wilson ([47]) point out, replacing $\mathbf{Z}$ with $\mathbf{W}$ introduces no additional approximation.

4.5 Convergence and Stopping Rules

The two parameters of the SEM algorithm described above are the number of SEM steps and the number of Gibbs iterations performed at each SEM step while generating $\mathbf{W}$ given $(\mathbf{X}, \boldsymbol{\theta}^{(n)})$. The choice of both parameters presents a problem.

Cowles and Carlin [14] and Brooks and Roberts [8] give a review of convergence diagnostics for MCMC. Raftery and Lewis [53] suggest methods for determining the number of steps required for the 'burn-in,' and for diagnosing lack of convergence or slow convergence, based on fitting first- and second-order Markov models to a sequence of every k-th iterations of the algorithm.

The stopping criterion usually adopted with the EM algorithm is in terms of either the size of the relative change in the parameter estimates or the log likelihood. As Lindstrom and Bates [39] emphasize, however, this is a measure of lack of progress but not of actual convergence. In our case (the SEM algoritm) this approach is further complicated by the Gibbs component and the approximation used when computing the expectation step.

In view of these difficulties, to decide on both stopping rules we prefer to follow a general recommendation contained in the roundtable discussion by Kass, Carlin, Gelman, and Neal [33] – namely, to make several runs of the algorithm from different starting points and to look at the trace plots of the components of $\boldsymbol{\theta}$ to confirm that each time they 'converge' to the same values.

4.6 Simulations

We have performed two simulations: one estimating the one-dimensional θ in the spherical family, and another one estimating the two-dimensional $\boldsymbol{\theta}$ in the exponential family. In both cases, 32×32 Gaussian images were obtained by the Circulant Embedding method (Section 3.3.2), and clipped at thresholds $\{\Phi^{-1}(1/3), \Phi^{-1}(2/3)\}$, assumed known. Three hundred steps of the SEM algorithm were performed on the clipped image. At each step, 30 iterations of the Gibbs sampler were used to generate $\mathbf{W} \mid \mathbf{X}, \boldsymbol{\theta}^{(n)}$.

The computational setup was the same as in Section 3.5. For the two-dimensional log likelihood maximization, a routine by Johnson [31] was used, which is derived from the Algol pseudocode in [32]. For the single-dimensional maximization we have used a simpler `fminbr` routine [36], which implements the "golden section" procedure combined with the parabolic interpolation, following [24].

Figures 4.9 and 4.10 show the SEM paths and the true parameter values. The precision of the estimators is discussed in the next chapter.

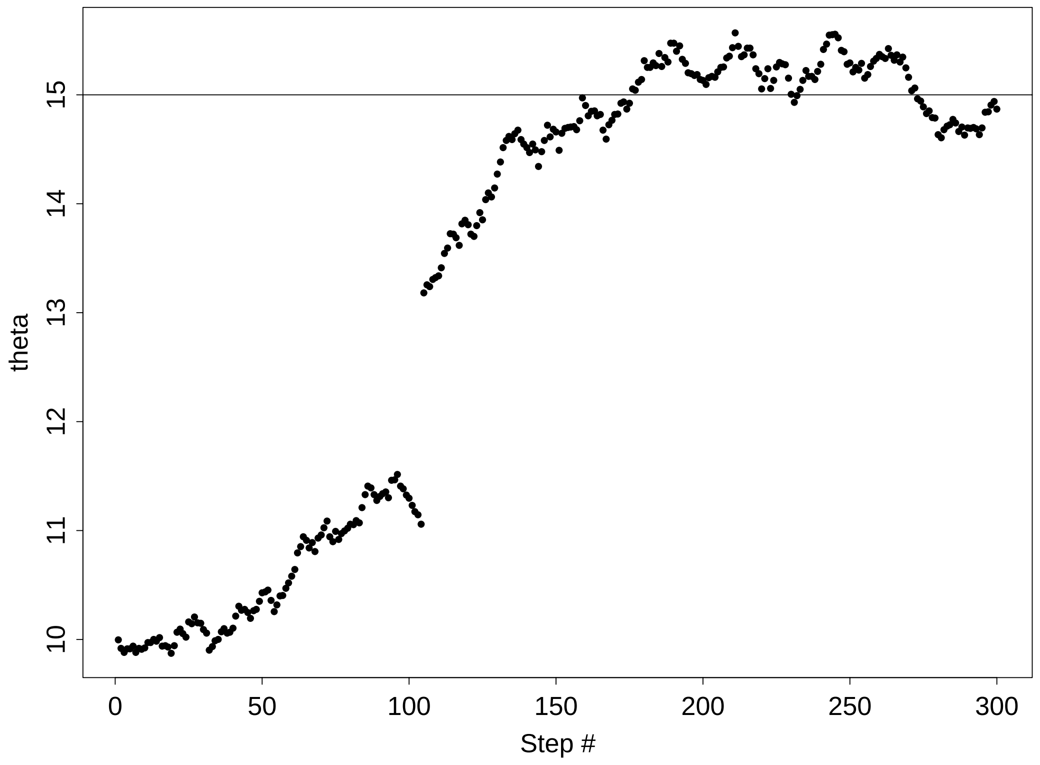

Figure 4.9: Three hundred steps of the SEM algorithm estimating θ from the 32×32 image obtained by clipping a Gaussian field with spherical covariance function at known levels $\{\Phi^{-1}(1/3), \Phi^{-1}(2/3)\}$. The true value is $\theta = 15$. The "discontinuity" of the SEM path is not uncommon for this setup.

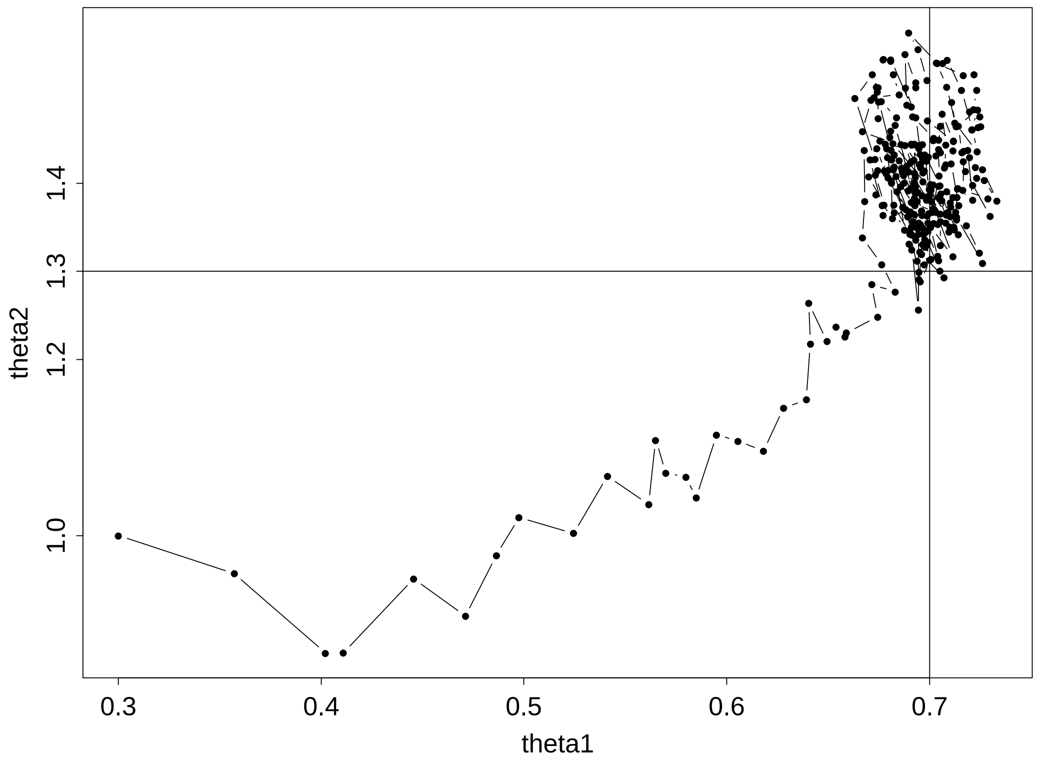

Figure 4.10: Three hundred steps of the SEM algorithm estimating $\boldsymbol{\theta}$ from the 32×32 image obtained by clipping a Gaussian field with exponentional covariance function at known levels $\{\Phi^{-1}(1/3), \Phi^{-1}(2/3)\}$. The true value is $\boldsymbol{\theta} = (0.7, 1.3)$.

Chapter 5

ESTIMATION FROM THE ORIGINAL AND FROM THE CLIPPED DATA

5.1 Introduction

We continue to study the model for discrete random fields introduced in the previous chapter, and investigate how much information about $\boldsymbol{\theta}$ is lost by quantization. We do this by comparing the variance of the estimators of $\boldsymbol{\theta}$ obtained from the original Gaussian data $\mathbf{Z}$ and from the clipped version $\mathbf{X}$. Estimation from the Gaussian data has been studied by Kitanidis [37], Kitanidis and Lane [38], Mardia and Marshall [41], Warnes and Ripley [62], and Mardia and Watkins [42].

5.2 Estimation from the original (Gaussian) data.

5.2.1 Maximum likelihood estimator

Suppose we have an observation $\mathbf{Z}$ of a stationary Gaussian field with mean zero and the covariance function $r_{\boldsymbol{\theta}}$ over a grid $\boldsymbol{S}$. As usual, $\mathbf{Z}$ is a $n = n_1 n_2$-vector obtained by writing down the $n_1 \times n_2$ image row-wise. Then $\mathbf{Z}$ is multivariate normal, $\mathbf{Z} \sim N(\mathbf{0}, \mathbf{T}_{\boldsymbol{\theta}})$. The parameter $\boldsymbol{\theta}$ can be estimated from $\mathbf{Z}$ by maximum likelihood:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \log L(\boldsymbol{\theta}, \mathbf{Z}),$$

where

$$\begin{aligned}
&\log L(\boldsymbol{\theta}, \mathbf{Z}) \\
&\quad = \log\left((2\pi)^{-n_1 n_2/2} |\mathbf{T}_{\boldsymbol{\theta}}|^{-1/2} \exp\left(-\mathbf{Z}^T \mathbf{T}_{\boldsymbol{\theta}}^{-1} \mathbf{Z}/2\right)\right) \\
&\quad = -\frac{1}{2}(\log |\mathbf{T}_{\boldsymbol{\theta}}| + \mathbf{Z}^T \mathbf{T}_{\boldsymbol{\theta}}^{-1} \mathbf{Z}) + C \qquad (5.1)
\end{aligned}$$

and

$$\mathbf{T}_{\boldsymbol{\theta}} = \left\{ r(\mathbf{s}_i, \mathbf{s}_j, \boldsymbol{\theta}) \right\}_{i,j=1}^{n_1 n_2}.$$

This solves the estimation problem, since we can maximize $\log L(\boldsymbol{\theta}, \mathbf{Z})$ numerically and obtain $\hat{\boldsymbol{\theta}}$. However, we want to explore the precision of $\hat{\boldsymbol{\theta}}$, and therefore we have to proceed analytically to compute the Fisher information matrix.

5.2.2 Fisher information matrix

Taking derivatives in (5.1) with respect to θ_j, we get

$$\frac{\partial}{\partial \theta_j} \log L(\boldsymbol{\theta}, \mathbf{Z}) = -\frac{1}{2} \left(\mathrm{Tr} \left[\mathbf{T}_{\boldsymbol{\theta}}^{-1} \frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j} \right] - \mathbf{Z}^T \left(\mathbf{T}_{\boldsymbol{\theta}}^{-1} \frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j} \mathbf{T}_{\boldsymbol{\theta}}^{-1} \right) \mathbf{Z} \right). \qquad (5.2)$$

where the following relations (Athans and Schweppe, [2], p. 15 and 28) were used:

$$\begin{aligned}
\frac{\partial}{\partial \theta_j} |\mathbf{T}_{\boldsymbol{\theta}}| &= |\mathbf{T}_{\boldsymbol{\theta}}| \mathrm{Tr} \left[\mathbf{T}_{\boldsymbol{\theta}}^{-1} \frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j} \right], \\
\frac{\partial}{\partial \theta_j} \mathbf{T}_{\boldsymbol{\theta}}^{-1} &= -\mathbf{T}_{\boldsymbol{\theta}}^{-1} \frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j} \mathbf{T}_{\boldsymbol{\theta}}^{-1}.
\end{aligned}$$

The second derivatives of $\log L(\boldsymbol{\theta}, \mathbf{Z})$ can be written in the following way:

$$\begin{aligned}
&E\left(\frac{\partial^2}{\partial\theta_j\partial\theta_k}\log L(\boldsymbol{\theta},\mathbf{Z})\right)\\
&\quad= E\left(\frac{\partial}{\partial\theta_j}\left(\frac{\partial}{\partial\theta_k}\log L(\boldsymbol{\theta},\mathbf{Z})\right)\right)\\
&\quad= E\left(\frac{\partial}{\partial\theta_j}\left(\frac{(\partial/\partial\theta_k)L(\boldsymbol{\theta},\mathbf{Z})}{L(\boldsymbol{\theta},\mathbf{Z})}\right)\right)\\
&\quad= E\left(\frac{((\partial^2/\partial\theta_j\partial\theta_k)L(\boldsymbol{\theta},\mathbf{Z}))L(\boldsymbol{\theta},\mathbf{Z}) - (\partial/\partial\theta_j)L(\boldsymbol{\theta},\mathbf{Z})(\partial/\partial\theta_k)L(\boldsymbol{\theta},\mathbf{Z})}{L(\boldsymbol{\theta},\mathbf{Z})^2}\right)\\
&\quad= E\left(\frac{(\partial^2/\partial\theta_j\partial\theta_k)L(\boldsymbol{\theta},\mathbf{Z})}{L(\boldsymbol{\theta},\mathbf{Z})}\right) - E\left(\frac{\partial}{\partial\theta_j}\log L(\boldsymbol{\theta},\mathbf{Z})\frac{\partial}{\partial\theta_k}\log L(\boldsymbol{\theta},\mathbf{Z})\right).
\end{aligned}$$

Since for exponential families we can differentiate under the integral sign, the first term in this expression is zero:

$$\begin{aligned}
&E\left(\frac{\frac{\partial^2}{\partial\theta_j\partial\theta_k}L(\boldsymbol{\theta},\mathbf{Z})}{L(\boldsymbol{\theta},\mathbf{Z})}\right)\\
&\quad= \int_{\mathbf{z}\in\mathbb{R}^{n_1n_2}}\frac{\frac{\partial^2}{\partial\theta_j\partial\theta_k}L(\boldsymbol{\theta},\mathbf{z})}{L(\boldsymbol{\theta},\mathbf{z})}L(\boldsymbol{\theta},\mathbf{z})d\mathbf{z}\\
&\quad= \frac{\partial^2}{\partial\theta_j\partial\theta_k}\int_{\mathbf{z}\in\mathbb{R}^{n_1n_2}}L(\boldsymbol{\theta},\mathbf{z})d\mathbf{z}\\
&\quad= \frac{\partial^2 1}{\partial\theta_j\partial\theta_k}\\
&\quad= 0.
\end{aligned}$$

Therefore, using (5.2) we have

$$\begin{aligned}
&E\left(\frac{\partial^2}{\partial\theta_j\partial\theta_k}\log L(\boldsymbol{\theta},\mathbf{Z})\right)\\
&\quad= -E\left(\frac{\partial}{\partial\theta_j}\log L(\boldsymbol{\theta},\mathbf{Z})\frac{\partial}{\partial\theta_k}\log L(\boldsymbol{\theta},\mathbf{Z})\right)\\
&\quad= -\frac{1}{4}E\left(\left(\operatorname{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_j}\right] - \mathbf{Z}^T\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_j}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\right)\right.\\
&\qquad\qquad\left.\times\left(\operatorname{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_k}\right] - \mathbf{Z}^T\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_k}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\right)\right).
\end{aligned}$$

For any matrix $\mathbf{A}$, it is true that $\mathbf{z}^T\mathbf{A}\mathbf{z} = \mathrm{Tr}(\mathbf{A}\mathbf{z}\mathbf{z}^T)$, since

$$\begin{aligned}\mathbf{z}^T\mathbf{A}\mathbf{z} &= \mathrm{Tr}(\mathbf{z}^T\mathbf{A}\mathbf{z}) \quad \text{(a scalar)}\\ &= \mathrm{Tr}(\mathbf{A}\mathbf{z}\mathbf{z}^T) \quad \text{(trace is invariant under cyclical permutations)}\end{aligned}$$

Therefore, we can rewrite the last expression as

$$\begin{aligned}&E\left(\frac{\partial^2}{\partial\theta_j\partial\theta_k}\log L(\boldsymbol{\theta},\mathbf{Z})\right)\\ &= -\frac{1}{4}E\left(\left(\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_j}\right] - \mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_j}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\mathbf{Z}^T\right]\right)\right.\\ &\qquad\left.\times\left(\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_k}\right] - \mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_k}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\mathbf{Z}^T\right]\right)\right)\\ &= -\frac{1}{4}E\left(\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_k}\right]\right.\\ &\qquad - \mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_j}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\mathbf{Z}^T\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_k}\right]\\ &\qquad - \mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_k}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\mathbf{Z}^T\right]\\ &\qquad\left. + \mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_j}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\mathbf{Z}^T\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_k}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\mathbf{Z}^T\right]\right)\end{aligned}$$

Because they are both being linear operations, trace and differentiation commute. Together with $E(\mathbf{Z}\mathbf{Z}^T) = \mathbf{T}_{\boldsymbol{\theta}}$, this allows us to proceed to

$$\begin{aligned}&E\left(\frac{\partial^2}{\partial\theta_j\partial\theta_k}\log L(\boldsymbol{\theta},\mathbf{Z})\right)\\ &= -\frac{1}{4}\left(\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_k}\right]\right.\\ &\qquad - \mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_j}\mathbf{T}_{\boldsymbol{\theta}}^{-1}E\left(\mathbf{Z}\mathbf{Z}^T\right)\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_k}\right]\\ &\qquad - \mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_k}\mathbf{T}_{\boldsymbol{\theta}}^{-1}E\left(\mathbf{Z}\mathbf{Z}^T\right)\right]\\ &\qquad\left. + E\left(\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_j}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\mathbf{Z}^T\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial\mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_k}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\mathbf{Z}^T\right]\right)\right)\end{aligned}$$

$$
\begin{aligned}
&= -\frac{1}{4}\Bigg(\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_k}\right] \\
&\qquad - \mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_k}\right] \\
&\qquad - \mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_k}\right] \\
&\qquad + E\left(\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\mathbf{Z}^T\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_k}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\mathbf{Z}^T\right]\right)\Bigg) \\
&= \frac{1}{4}\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_k}\right] \\
&\quad - \frac{1}{4}E\left(\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\mathbf{Z}^T\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_k}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}\mathbf{Z}^T\right]\right) \\
&= \frac{1}{4}\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_k}\right] \\
&\quad - \frac{1}{4}E\left(\left\{\mathbf{Z}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}^T\right\}\left\{\mathbf{Z}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_k}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\mathbf{Z}^T\right\}\right) \\
&= \frac{1}{4}\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_k}\right] \\
&\quad - \frac{1}{4}E\left(\sum_{a,b=1}^{n_1n_2}\left(\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\right)_{ab}Z_aZ_b\sum_{c,d=1}^{n_1n_2}\left(\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_k}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\right)_{cd}Z_cZ_d\right) \\
&= \frac{1}{4}\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_k}\right] \\
&\quad - \frac{1}{4}E\left(\sum_{a,b,c,d=1}^{n_1n_2}\left(\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\right)_{ab}\left(\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_k}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\right)_{cd}Z_aZ_bZ_cZ_d\right). \qquad (5.3)
\end{aligned}
$$

To compute $E(Z_aZ_bZ_cZ_d)$ we use the following lemma:

Lemma 2 *For* $\mathbf{Z} \sim N(\mathbf{0}, \mathbf{T})$ *and any* $1 \leq a, b, c, d \leq n_1n_2$,

$$E(Z_aZ_bZ_cZ_d) = E(Z_aZ_b)E(Z_cZ_d) + E(Z_aZ_c)E(Z_bZ_d) + E(Z_aZ_d)E(Z_bZ_d).$$

Proof: Consider the multivariate normal vector $(z_1, z_2, z_3, z_4) \equiv (Z_a, Z_b, Z_c, Z_d)$. Its moment generating function is given by

$$M(t_1, t_2, t_3, t_4) = E\exp(t_1z_1 + \cdots + t_4z_4)$$

$$\begin{aligned}
&= \exp\left(\frac{1}{2}\sum_{i,j=1}^{4} \mathrm{Cov}(z_i, z_j) t_i t_j\right) \\
&= \exp\bigg(\frac{1}{2}\sigma_{11}t_1^2 + \frac{1}{2}\sigma_{22}t_2^2 + \frac{1}{2}\sigma_{33}t_3^2 + \frac{1}{2}\sigma_{44}t_4^2 \\
&\qquad + \sigma_{12}t_1t_2 + \sigma_{13}t_1t_3 + \sigma_{14}t_1t_4 \\
&\qquad + \sigma_{23}t_2t_3 + \sigma_{24}t_2t_4 \\
&\qquad + \sigma_{34}t_2t_4\bigg),
\end{aligned}$$

where $\sigma_{ij} = \mathrm{Cov}(z_i, z_j)$ (see Priestley [52], p. 91). The fourth moment can be computed by evaluating the fourth mixed derivative of M at zero:

$$E(z_1z_2z_3z_4) = \left.\frac{\partial^4 M(t_1,t_2,t_3,t_4)}{\partial t_1\partial t_2\partial t_3\partial t_4}\right|_{(t_1,t_2,t_3,t_4)=\mathbf{0}}.$$

Differentiating, we get

$$\frac{\partial M(\mathbf{t})}{\partial t_1} = (\sigma_{11}t_1 + \sigma_{12}t_2 + \sigma_{13}t_3 + \sigma_{14}t_4)M(\mathbf{t}).$$

$$\begin{aligned}
\frac{\partial^2 M(\mathbf{t})}{\partial t_1\partial t_2} &= \sigma_{12}M(\mathbf{t}) + (\sigma_{11}t_1 + \sigma_{12}t_2 + \sigma_{13}t_3 + \sigma_{14}t_4) \\
&\qquad \times (\sigma_{22}t_2 + \sigma_{12}t_1 + \sigma_{23}t_3 + \sigma_{24}t_4)M(\mathbf{t}).
\end{aligned}$$

$$\begin{aligned}
\frac{\partial^3 M(\mathbf{t})}{\partial t_1\partial t_2\partial t_3} =& \sigma_{12}(\sigma_{33}t_3 + \sigma_{13}t_1 + \sigma_{23}t_2 + \sigma_{34}t_4)M(\mathbf{t}) \\
&+ \sigma_{13}(\sigma_{22}t_2 + \sigma_{12}t_1 + \sigma_{23}t_3 + \sigma_{24}t_4)M(\mathbf{t}) \\
&+ (\sigma_{11}t_1 + \sigma_{12}t_2 + \sigma_{13}t_3 + \sigma_{14}t_4)\sigma_{23}M(\mathbf{t}) \\
&+ (\sigma_{11}t_1 + \sigma_{12}t_2 + \sigma_{13}t_3 + \sigma_{14}t_4) \\
&\qquad \times (\sigma_{22}t_2 + \sigma_{12}t_1 + \sigma_{23}t_3 + \sigma_{24}t_4) \\
&\qquad\qquad \times (\sigma_{33}t_3 + \sigma_{13}t_1 + +\sigma_{23}t_2 + \sigma_{34}t_4)M(\mathbf{t}).
\end{aligned}$$

$$\left.\frac{\partial^4 M(\mathbf{t})}{\partial t_1\partial t_2\partial t_3\partial t_4}\right|_{\mathbf{t}=\mathbf{0}} = \sigma_{12}\sigma_{34} + \sigma_{13}\sigma_{24} + \sigma_{14}\sigma_{23}.$$

Therefore, switching back to the original notation,

$$\begin{aligned}E(Z_aZ_bZ_cZ_d) =&\text{Cov}(Z_a, Z_b)\text{Cov}(Z_c, Z_d) + \\ &+ \text{Cov}(Z_a, Z_c)\text{Cov}(Z_b, Z_d) + \\ &+ \text{Cov}(Z_a, Z_d)\text{Cov}(Z_b, Z_c),\end{aligned}$$

which concludes the proof of Lemma 2.

The following fact will also be used:

Lemma 3 *For any two $n \times n$ matrices $\mathbf{A}$ and $\mathbf{B}$,*

$$Tr(\mathbf{A}\mathbf{B}^T) = \sum_{i,j=1}^{n} A_{ij}B_{ij}.$$

Proof.

$$\text{Tr}(\mathbf{A}\mathbf{B}^T) = \sum_{i=1}^{n}(\mathbf{A}\mathbf{B}^T)_{ii} = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}(B^T)_{ji} = \sum_{i,j=1}^{n} A_{ij}B_{ij}.$$

We now continue the computation of the Fisher information matrix. The second term in (5.3) becomes

$$\begin{aligned}&\frac{1}{4}E\left(\sum_{a,b,c,d=1}^{n_1n_2}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)_{ab}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)_{cd}Z_aZ_bZ_cZ_d\right)\\
&\quad= \frac{1}{4}\sum_{a,b,c,d=1}^{n_1n_2}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)_{ab}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)_{cd}E(Z_aZ_bZ_cZ_d)\\
&\quad= \frac{1}{4}\sum_{a,b,c,d=1}^{n_1n_2}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)_{ab}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)_{cd}\\
&\qquad\qquad\times\left(T_{\boldsymbol\theta ab}T_{\boldsymbol\theta cd} + T_{\boldsymbol\theta ac}T_{\boldsymbol\theta bd} + T_{\boldsymbol\theta ad}T_{\boldsymbol\theta bc}\right)\quad\text{(see Lemma 2)}\\
&\quad= \frac{1}{4}\sum_{a,b=1}^{n_1n_2}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)_{ab}T_{\boldsymbol\theta ab}\sum_{c,d=1}^{n_1n_2}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)_{cd}T_{\boldsymbol\theta cd}\\
&\qquad+\frac{1}{2}\sum_{a,b=1}^{n_1n_2}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)_{ab}\sum_{c,d=1}^{n_1n_2}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)_{cd}T_{\boldsymbol\theta ac}T_{\boldsymbol\theta bd}\end{aligned}$$

(because c and d are interchangeable by symmetry)

$$= \frac{1}{4}\mathrm{Tr}\left[\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)\mathbf{T}_{\boldsymbol\theta}\right]\mathrm{Tr}\left[\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)\mathbf{T}_{\boldsymbol\theta}\right] \quad \text{(see Lemma 3)}$$

$$+\frac{1}{2}\sum_{a,b=1}^{n_1n_2}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)_{ab}(\mathbf{T}_{\boldsymbol\theta}^{T})_a\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)(\mathbf{T}_{\boldsymbol\theta})_b$$

$$= \frac{1}{4}\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\right]$$

$$+\frac{1}{2}\sum_{a,b=1}^{n_1n_2}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)_{ab}\left(\mathbf{T}_{\boldsymbol\theta}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)\mathbf{T}_{\boldsymbol\theta}\right)_{ab}$$

$$= \frac{1}{4}\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\right]$$

$$+\frac{1}{2}\mathrm{Tr}\left[\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)\mathbf{T}_{\boldsymbol\theta}\left(\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\mathbf{T}_{\boldsymbol\theta}^{-1}\right)\mathbf{T}_{\boldsymbol\theta}\right]$$

$$= \frac{1}{4}\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\right]\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\right]+\frac{1}{2}\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\right].$$

Substituting this expression into (5.3) establishes the following theorem:

Theorem 2 *The (j,k)-th element of the Fisher information matrix for $\boldsymbol\theta$ in the model $\mathbf{Z}\sim N(\mathbf{0},\mathbf{T}_{\boldsymbol\theta})$ is given by*

$$M_{jk} = -E\left(\frac{\partial^2}{\partial\theta_j\partial\theta_k}\log L(\boldsymbol\theta,\mathbf{Z})\right) = \frac{1}{2}Tr\left[\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_j}\mathbf{T}_{\boldsymbol\theta}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol\theta}}{\partial\theta_k}\right]. \qquad (5.4)$$

5.2.3 Asymptotic normality and efficiency of the maximum likelihood estimator

Note that $\mathbf{Z}$ is a single observation of the field. Therefore, it is not immediate that $\hat{\boldsymbol\theta}$ obtained by the maximum likelihood method is consistent and asymptotically normal (as the grid expands). The properties of the maximum likelihood estimator for dependent data in the general case are discussed among other places in Billingsley [4], Bhat [3], Crowder [16], and Sweeting [58].

Our starting point is the following theorem (Theorem 2 in Mardia and Marshall [41]):

Theorem 3 *Suppose that the n-vector $\mathbf{Z}$ is a single observation from $N(\mathbf{0}, \mathbf{T}_{\boldsymbol{\theta}})$. The length of $\boldsymbol{\theta}$ is p. Let $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$ denote the eigenvalues of $\mathbf{T}_{\boldsymbol{\theta}}$, let $\lambda_1^i \le \lambda_2^i \le \cdots \le \lambda_n^i$ denote the eigenvalues of $\partial \mathbf{T}_{\boldsymbol{\theta}}/\partial\theta_i$, and let $\lambda_1^{ij} \le \lambda_2^{ij} \le \cdots \le \lambda_n^{ij}$ denote the eigenvalues of $\partial^2 \mathbf{T}_{\boldsymbol{\theta}}/\partial\theta_i\partial\theta_j$. Suppose also that the following conditions hold.*

1. $\lim_{n\to\infty} \lambda_n = C < \infty$,

   $\lim_{n\to\infty} |\lambda_n^i| = C_i < \infty$,

   $\lim_{n\to\infty} |\lambda_n^{ij}| = C_{ij} < \infty$;

2. $\|\partial \mathbf{T}_{\boldsymbol{\theta}}/\partial\theta_i\|_F^{-2} = O(n^{-\frac{1}{2}-\delta})$, *for some* $\delta > 0$ *and for* $i = 1, \ldots, p$;
   here $\|.\|_F$ *denotes the Frobenius matrix norm,* $\|\mathbf{A}\|_F = \sqrt{\sum_i \sum_j |A_{ij}|^2}$.

3. *For all* $i, j = 1, \ldots, p$, *define*

$$t_{ij} = Tr\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_i}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta_j}\right].$$

   Then we assume the limit

$$A_{ij} = \lim_{n\to\infty} \frac{t_{ij}}{\sqrt{t_{ii}t_{jj}}}.$$

   exists and $\mathbf{A} = (A_{ij})$ *is a non-singular matrix.*

Then the maximum likelihood estimator of $\boldsymbol{\theta}$ obtained from $\mathbf{Z}$ is asymptotically normal; that is, $\hat{\boldsymbol{\theta}} \sim N(\boldsymbol{\theta}, \mathbf{M}^{-1})$, where the Fisher information matrix $\boldsymbol{M}$ is given by (5.4).

Following in part the same paper (Mardia and Marshall [41]), this theorem can be specialized to our case of a sample from a stationary field over a regular grid, and also to the encompassing (in the sense of circulant embedding method) sample, to give simple conditions on the covariance function $r_{\boldsymbol{\theta}}$. Namely, the following holds:

Theorem 4 *Suppose that $r(\mathbf{k}, \boldsymbol{\theta})$ is a covariance function twice continuously differentiable in $\boldsymbol{\theta}$. The length of $\boldsymbol{\theta}$ is p, and $\mathbf{k} \in \mathbb{Z}^2$ is a two-dimensional lag. We fix a regular $n_1 \times n_2$ grid with $n = n_1 n_2$ points and define $\mathbf{T}_{\boldsymbol{\theta}}$ to be the covariance matrix of a single observation of a stationary zero-mean Gaussian field with the covariance function r over this grid. We also define $\mathbf{C}_{\boldsymbol{\theta}}$ to be the block circulant matrix encompassing $\mathbf{T}_{\boldsymbol{\theta}}$. Assume that $\mathbf{C}_{\boldsymbol{\theta}}$ is non-negative definite and that the following conditions are satisfied.*

1. *For all $i, j = 1, \ldots, p$, define*

$$t_{ij} = Tr\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1} \frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_i} \mathbf{T}_{\boldsymbol{\theta}}^{-1} \frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial \theta_j}\right].$$

   Then we assume the limit

$$A_{ij} = \lim_{n \to \infty} \frac{t_{ij}}{\sqrt{t_{ii} t_{jj}}}$$

   exists and $\mathbf{A} = (A_{ij})$ is a non-singular matrix.

2. *For all $i, j = 1, \ldots, p$, the three following series are absolutely summable:*

$$\sum_{\mathbf{k} \in \mathbb{Z}^2} |r(\mathbf{k}, \boldsymbol{\theta})| < \infty,$$
$$\sum_{\mathbf{k} \in \mathbb{Z}^2} \left| \frac{\partial}{\partial \theta_i} r(\mathbf{k}, \boldsymbol{\theta}) \right| < \infty, \text{ and}$$
$$\sum_{\mathbf{k} \in \mathbb{Z}^2} \left| \frac{\partial^2}{\partial \theta_i \partial \theta_j} r(\mathbf{k}, \boldsymbol{\theta}) \right| < \infty.$$

Then the maximum likelihood estimators of $\boldsymbol{\theta}$ obtained from $\mathbf{Z} \sim N(0, \mathbf{T}_{\boldsymbol{\theta}})$ and from $\mathbf{W} \sim N(0, \mathbf{C}_{\boldsymbol{\theta}})$ are asymptotically normal; that is, $\hat{\boldsymbol{\theta}} \sim N(\boldsymbol{\theta}, \mathbf{M}^{-1})$, $n \to \infty$.

The Fisher information matrix $\boldsymbol{M}$ *is given by* (5.4) *for estimation from* $\mathbf{Z}$*, and by analogous matrix with* $\mathbf{T}_{\boldsymbol{\theta}}$ *replaced with* $\mathbf{C}_{\boldsymbol{\theta}}$ *in* (5.4) *for estimation from* $\mathbf{W}$*.*

Proof: We will verify the conditions of Theorem 3.

From matrix norm properties, the spectral radii of $\mathbf{T}_{\boldsymbol{\theta}}$ and $\mathbf{C}_{\boldsymbol{\theta}}$, that is, λ_n^T and λ_n^C, are no larger than the row sum norms of $\mathbf{T}_{\boldsymbol{\theta}}$ and $\mathbf{C}_{\boldsymbol{\theta}}$ respectively (Stoer and Bulirsch [57], p.406, and Golub and van Loan [28], p.56), defined as

$$\|\mathbf{A}\|_\infty = \max_i \sum_j |A_{ij}|.$$

As the grid expands, $\|\mathbf{T}_{\boldsymbol{\theta}}\|_\infty$ converges to the sum of $|r(\mathbf{k}, \boldsymbol{\theta})|$ over $\mathbf{k}$. Therefore, $\lim \lambda_n^T < \infty$ in condition 1 of Theorem 3 is ensured if $r_{\boldsymbol{\theta}}$ is absolutely summable over $\mathbb{Z}^2$.

Speaking of the encompassing matrix, $\mathbf{C}_{\boldsymbol{\theta}}$, note that all its row sums are equal because of the circulant structure; therefore it suffices to consider the first row only. By construction, it consists of the elements of the first row of $\mathbf{T}_{\boldsymbol{\theta}}$ (some of them repeated twice), the elements of the first row of the last block row of $\mathbf{T}_{\boldsymbol{\theta}}$ (some of them also repeated twice), and filler zeros. Therefore,

$$\|\mathbf{C}_{\boldsymbol{\theta}}\|_\infty \leq 4\|\mathbf{T}_{\boldsymbol{\theta}}\|_\infty,$$

and convergence of λ_n^C is also ensured.

Similarly, the remaining parts of condition 1 of Theorem 3 hold if $\partial r(\mathbf{k}, \boldsymbol{\theta})/\partial\theta_i$ and $\partial^2 r(\mathbf{k}, \boldsymbol{\theta})/\partial\theta_i\partial\theta_j$ are absolutely summable in $\mathbf{k}$.

Consider now condition 2 of Theorem 3. If we introduce notation

$$\sigma_{l,m,i} = \frac{\partial r((l,m), \boldsymbol{\theta})}{\partial\theta_i},$$

then, in view of the structure of $\mathbf{T}_{\boldsymbol{\theta}}$,

$$\|\partial\mathbf{T}_{\boldsymbol{\theta}}/\partial\theta_i\|_F^2 = \sum_{k_1=0}^{n_1-1}(n_1 - k_1)\sum_{k_2=0}^{n_2-1}(n_2 - k_2)\sigma_{k_1,k_2,i}^2$$

$$\begin{aligned}
&= \sum_{k_1=0}^{n_1-1} \sum_{k_2=0}^{n_2-1} (n_1 n_2 - k_1 n_2 - n_1 k_2 + k_1 k_2) \sigma^2_{k_1,k_2,i} \\
&= n_1 n_2 \sum_{k_1=0}^{n_1-1} \sum_{k_2=0}^{n_2-1} \sigma^2_{k_1,k_2,i} - n_2 \sum_{k_1=0}^{n_1-1} k_1 \sum_{k_2=0}^{n_2-1} \sigma^2_{k_1,k_2,i} \\
&\qquad - n_1 \sum_{k_1=0}^{n_1-1} \sum_{k_2=0}^{n_2-1} k_2 \sigma^2_{k_1,k_2,i} \\
&\qquad + \sum_{k_1=0}^{n_1-1} k_1 \sum_{k_2=0}^{n_2-1} k_2 \sigma^2_{k_1,k_2,i}.
\end{aligned}$$

After dividing both sides by $n = n_1 n_2$ we get

$$\begin{aligned}
\frac{1}{n} \|\partial \mathbf{T}_{\boldsymbol{\theta}} / \partial \theta_i\|_F^2 = \sum_{k_1=0}^{n_1-1} \sum_{k_2=0}^{n_2-1} \sigma^2_{k_1,k_2,i} &- \frac{1}{n_1} \sum_{k_1=0}^{n_1-1} k_1 \sum_{k_2=0}^{n_2-1} \sigma^2_{k_1,k_2,i} \\
&- \frac{1}{n_2} \sum_{k_1=0}^{n_1-1} \sum_{k_2=0}^{n_2-1} k_2 \sigma^2_{k_1,k_2,i} \\
&+ \frac{1}{n} \sum_{k_1=0}^{n_1-1} k_1 \sum_{k_2=0}^{n_2-1} k_2 \sigma^2_{k_1,k_2,i}. \qquad (5.5)
\end{aligned}$$

We claim that all the terms in the right hand side of (5.5) but the first one converge to zero as $n_1, n_2 \to \infty$. To show this we use Kronecker's lemma: *if (a_s) and (b_s) are real sequences with $b_s \to \infty$, and $\sum_s a_s/b_s < \infty$, then $b_s^{-1} \sum_{j=1}^{s} a_j \to 0$ as $s \to \infty$.*

Since $\sigma_{k_1,k_2,i}$ is absolutely summable by assumption, so is $\sigma^2_{k_1,k_2,i}$. In particular, if we set

$$\begin{aligned}
a_s &= s \sum_{k_2=0}^{n_2-1} \sigma^2_{s,k_2,i}, \\
b_s &= s,
\end{aligned}$$

then by applying Kronecker's lemma to the second term in (5.5),

$$\frac{1}{n_1} \sum_{k_1=0}^{n_1-1} a_{k_1} = \frac{1}{n_1} \sum_{k_1=0}^{n_1-1} k_1 \sum_{k_2=0}^{n_2-1} \sigma^2_{k_1,k_2,i} \to 0 \quad \text{as} \quad n_1 \to \infty.$$

After changing the order of the summation, we can analyze the third term in (5.5) similarly:

$$\frac{1}{n_2}\sum_{k_1=0}^{n_1-1}\sum_{k_2=0}^{n_2-1} k_2\sigma^2_{k_1,k_2,i} \to 0 \quad \text{as} \quad n_2 \to \infty.$$

For the fourth term, we have

$$\begin{aligned}\frac{1}{n_1}\frac{1}{n_2}\sum_{k_1=0}^{n_1-1} k_1 \sum_{k_2=0}^{n_2-1} k_2\sigma^2_{k_1,k_2,i} \\ &= \frac{1}{n_1}\sum_{k_1=0}^{n_1-1} k_1 \sum_{k_2=0}^{n_2-1} \frac{k_2}{n_2}\sigma^2_{k_1,k_2,i} \\ &\le \frac{1}{n_1}\sum_{k_1=0}^{n_1-1} k_1 \sum_{k_2=0}^{n_2-1} \sigma^2_{k_1,k_2,i} \to 0 \quad \text{as} \quad n_2 \to \infty.\end{aligned}$$

Therefore, if we take limits on both sides of (5.5), we get

$$\lim \frac{1}{n}\|\partial \mathbf{T}_{\boldsymbol{\theta}}/\partial\theta_i\|_F^2 = \sum_{\mathbf{k}} \sigma^2_{\mathbf{k},i} < \infty,$$

which, upon inversion, shows that condition 2 of Theorem 3 holds with $\delta = 1/2$.

Since

$$|\partial \mathbf{C}_{\boldsymbol{\theta}}/\partial\theta_i\|_F^2 \le 16|\partial \mathbf{T}_{\boldsymbol{\theta}}/\partial\theta_i\|_F^2,$$

the same holds for the encompassing sample case.

Condition 3 of Theorem 3 is assumed, which concludes the proof of Theorem 4.

■

Note that in practice, we would normally have **Z** rather than **W** as data; however, we use **W** for simulations.

The conditions of Theorem 4 are easy to check for the spherical covariance function

$$r(\mathbf{k},\theta) = \begin{cases} 1 - \frac{3}{2}\left(\frac{\|\mathbf{k}\|}{\theta}\right) + \frac{1}{2}\left(\frac{\|\mathbf{k}\|}{\theta}\right)^3, & \|\mathbf{k}\| \le \theta, \\ 0, & \text{otherwise}, \end{cases} \tag{5.6}$$

where $\theta > 0$ and $\|\mathbf{k}\| = \sqrt{k_1^2 + k_2^2}$. For the first condition, we have

$$A_{11} = \lim_{n\to\infty} \frac{t_{11}}{\sqrt{t_{11}t_{11}}} = 1.$$

The second condition also holds, since in each of the three series there is only a finite number of non-zero terms. Therefore, from Theorems 2 and 4 we have the theoretical asymptotic variance for the maximum likelihood estimator of θ:

$$\mathrm{Var}(\hat{\theta}) \approx 2\left(\mathrm{Tr}\left[\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta}\mathbf{T}_{\boldsymbol{\theta}}^{-1}\frac{\partial \mathbf{T}_{\boldsymbol{\theta}}}{\partial\theta}\right]\right)^{-1}. \tag{5.7}$$

5.2.4 Simulations

We performed simulations to find out how large n should be in order for (5.7) to work in the spherical covariance case. As noted above, the block circulant version (that is, $\mathbf{W}$ rather than $\mathbf{Z}$) was used to utilize the matrix factorization described in Section 2.6.

For the spherical covariance function (5.6),

$$\frac{\partial}{\partial\theta} r(\mathbf{k},\theta) = \begin{cases} \frac{3}{2}\frac{\|\mathbf{k}\|}{\theta^2}\left(1 - \frac{\|\mathbf{k}\|^2}{\theta^2}\right), & \|\mathbf{k}\| < \theta \\ 0, & \text{otherwise}. \end{cases} \tag{5.8}$$

Obviously, the matrix $\partial\mathbf{C}_\theta/\partial\theta$ has the same block circulant structure as $\mathbf{C}_\theta$. Let $\boldsymbol{\Lambda}$ and $\mathbf{M}$ denote diagonal matrices of eigenvalues of $\mathbf{C}_\theta$ and $\partial\mathbf{C}_\theta/\partial\theta$ respectively, and $\mathbf{F}$ be the two-dimensional Fourier transform matrix. Then the trace in (5.7) can be computed as

$$\begin{aligned}
&\operatorname{Tr}\left[\mathbf{C}_\theta^{-1}\frac{\partial \mathbf{C}_\theta}{\partial\theta}\mathbf{C}_\theta^{-1}\frac{\partial \mathbf{C}_\theta}{\partial\theta}\right] \\
&\quad = \operatorname{Tr}\left[\left(\mathbf{F}^H\mathbf{\Lambda}^{-1}\mathbf{F}\right)\left(\mathbf{F}^H\mathbf{M}\mathbf{F}\right)\left(\mathbf{F}^H\mathbf{\Lambda}^{-1}\mathbf{F}\right)\left(\mathbf{F}^H\mathbf{M}\mathbf{F}\right)\right] \\
&\quad = \operatorname{Tr}\left[\mathbf{F}^H\left(\mathbf{\Lambda}^{-1}\mathbf{M}\mathbf{\Lambda}^{-1}\mathbf{M}\right)\mathbf{F}\right] \\
&\quad = \operatorname{Tr}\left[\mathbf{\Lambda}^{-1}\mathbf{M}\mathbf{\Lambda}^{-1}\mathbf{M}\right] \\
&\quad = \sum_i \frac{\mu_i^2}{\lambda_i^2},
\end{aligned}$$

because the trace is invariant under orthogonal transformations. As we know, the eigenvalues λ_i and μ_i are readily available as two-dimensional Fourier transforms of the first columns of $\mathbf{C}_\theta$ and $\partial\mathbf{C}_\theta/\partial\theta$.

Figures 5.1 and 5.2 show the precision of the approximation (5.7) for various data sizes. The parameter θ is fixed at $\theta = 5$. The solid line is the graph of the right hand side computed by the above method. The dotted line was obtained as follows. For each value of n, 500 independent realizations $\mathbf{W}$ from $N(\mathbf{0}_n, \mathbf{C}_{\theta=5})$ were drawn (by the Circulant Embedding method described in Section 3.3.2), from each realization the MLE estimator $\hat{\theta}$ was obtained, and the sample variance of these 500 estimates was computed. The dotted line connects the sample variances. We see from Figure 5.2 that starting with the data size of one thousand points the fit becomes almost perfect.

5.3 Estimation from the clipped data

We now study how much information is lost when we quantize the Gaussian data and use the SEM algorithm for estimation of θ as in the model introduced in Section 4.3. We expect the estimation precision to increase with the number of

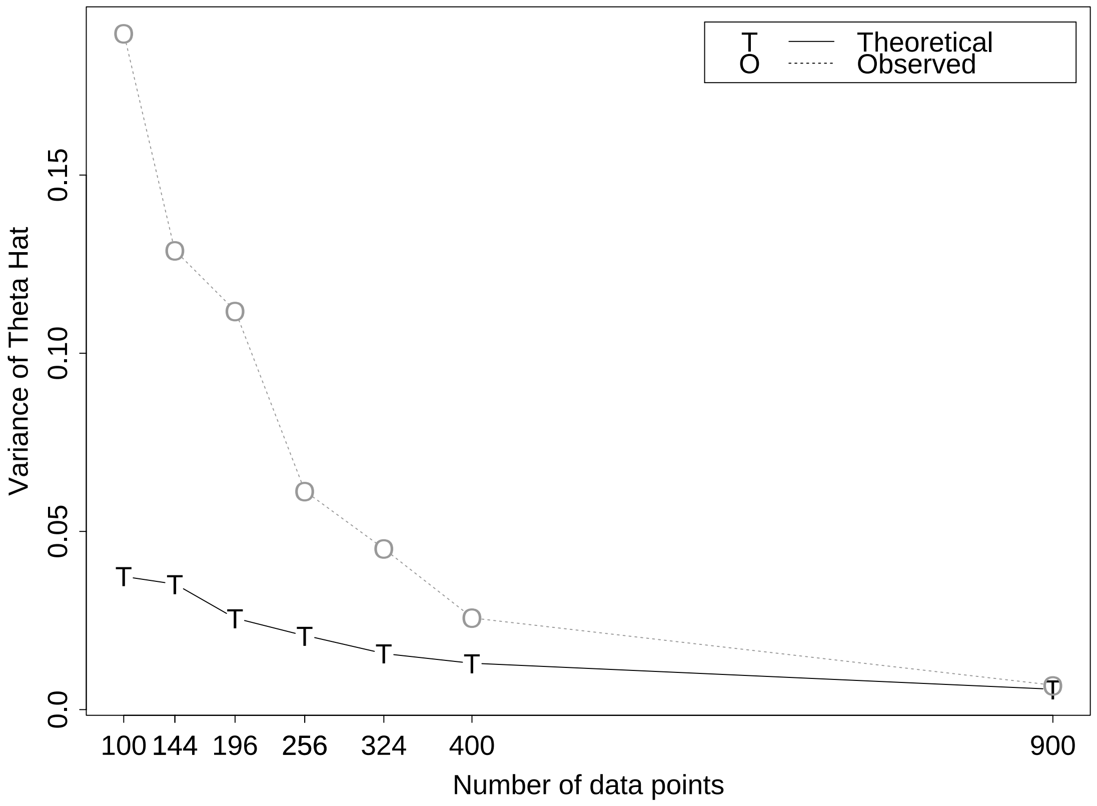

Figure 5.1: Variance of the MLE $\hat{\theta}$ as predicted by Theorem 2 and as observed in simulations. See text for explanation.

quantization levels, and approach in the limit the precision of estimation from the Gaussian data, described in the previous section.

Two series of simulations were performed. The first one used the same particular case of the spherical covariance function with the true value of $\theta = 5$. The second one used Matérn correlation with the second parameter fixed at value 0.1, and estimated the first parameter (having the true value of 2.1).

In a setup analogous to that of Figures 5.1 and 5.2, for each data vector size n we generated 250 independent realizations of $\mathbf{W} \sim N(\mathbf{0}_n, \mathbf{C}_{\theta=5})$. We then

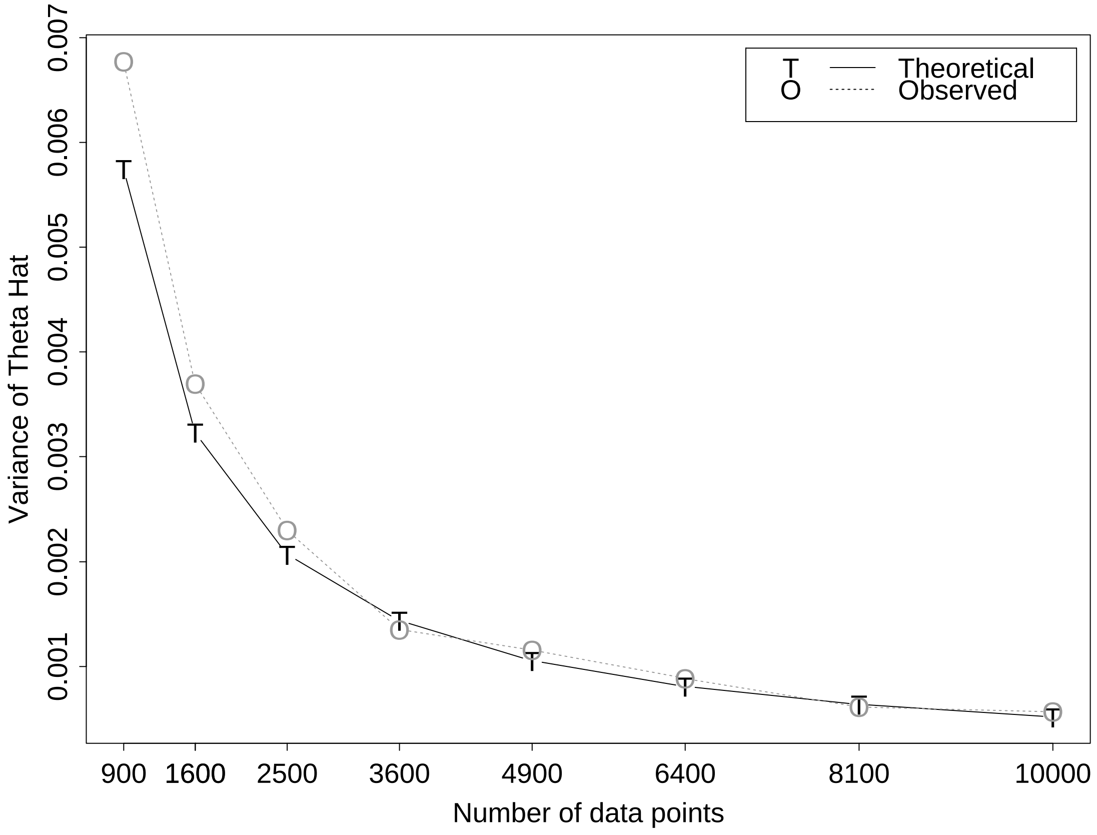

Figure 5.2: Continued from Figure 5.1.

quantized them at one, two, three, and five thresholds, obtaining binary, three-level, four-level and six-level discrete vectors respectively. On each of those we performed 150 steps of the SEM algorithm with 15 Gibbs iterations per step (see Section 4.4), and in each case recorded $\theta^{(150)}$. From these 250 estimates we computed and recorded the sample mean and variance of the estimates for the given data size and number of levels, and then moved on to the next data size.

The resulting graphs are shown in Figures 5.3 – 5.6 and in Figures 5.7 – 5.10. Analyzing the graphs of the mean squared error multiplied by the number of data points and noting that this graphs appear flat, we conjecture C/n behaviour of the MSE.

We also note that transition from the binary to the three-level quantization produces the largest improvement, and that for large values of n estimators from all quantizations perform reasonably well.

To compare the precision of different estimators, we consider the ratio of variances summarized in Table 5.1 and Table 5.2, where $\hat{\theta}_{(MLE)}$ is the maximum likelihood estimate from the Gaussian data, and $\hat{\theta}_{(k)}$ denotes the SEM estimate from the k-level discrete data.

These results are analogous to those for binary time series (Kedem [34], p. 60).

n	$\text{Var}\hat{\theta}_{(2)}/\text{Var}\hat{\theta}_{(MLE)}$	$\text{Var}\hat{\theta}_{(3)}/\text{Var}\hat{\theta}_{(MLE)}$	$\text{Var}\hat{\theta}_{(4)}/\text{Var}\hat{\theta}_{(MLE)}$	$\text{Var}\hat{\theta}_{(6)}/\text{Var}\hat{\theta}_{(MLE)}$
400	37.2	13.6	8.0	6.1
484	36.0	13.0	5.6	2.1
576	17.7	4.5	3.0	1.1
784	18.8	8.4	3.8	2.4
900	21.1	5.7	3.5	2.1
1024	18.5	5.1	3.5	2.3
1156	24.9	6.1	3.9	2.3
1444	20.8	4.3	2.8	1.9
1600	18.6	4.2	3.9	2.2

Table 5.1: Relative precision of estimators, Spherical(5) case. See also Figure 5.4.

n	$\text{Var}\hat{\theta}_{(2)}/\text{Var}\hat{\theta}_{(MLE)}$	$\text{Var}\hat{\theta}_{(3)}/\text{Var}\hat{\theta}_{(MLE)}$	$\text{Var}\hat{\theta}_{(4)}/\text{Var}\hat{\theta}_{(MLE)}$	$\text{Var}\hat{\theta}_{(6)}/\text{Var}\hat{\theta}_{(MLE)}$
400	3.9	2.9	2.0	2.0
484	3.7	1.9	1.9	1.4
576	4.3	2.6	2.1	1.7
784	6.7	2.5	2.4	1.6
900	5.3	3.0	1.8	1.4
1024	3.4	2.7	1.4	1.4
1156	4.6	2.4	1.5	1.2
1444	4.5	1.9	1.7	1.3
1600	5.7	2.5	2.1	1.8

Table 5.2: Relative precision of estimators, Matérn(2.1, 0.1) case. See also Figure 5.8.

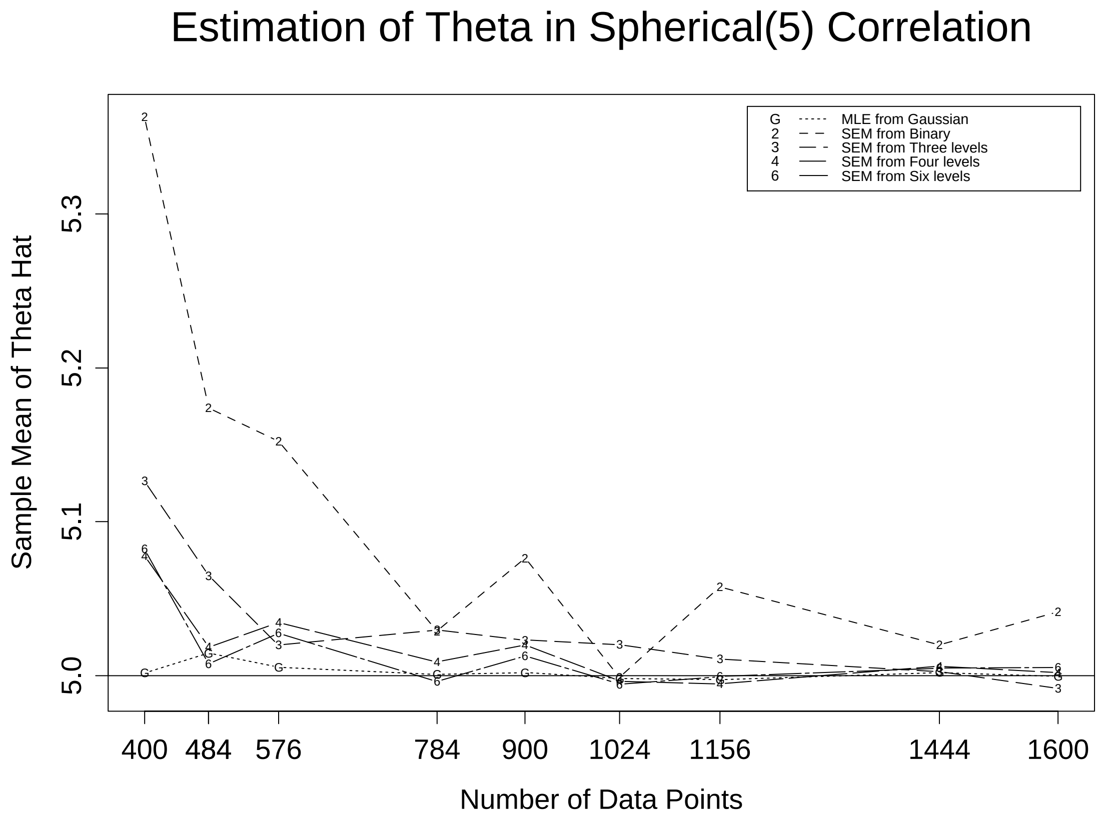

Figure 5.3: Mean of $\hat{\theta}$ when using MLE from the Gaussian data and SEM from the same data clipped at various number of thresholds.

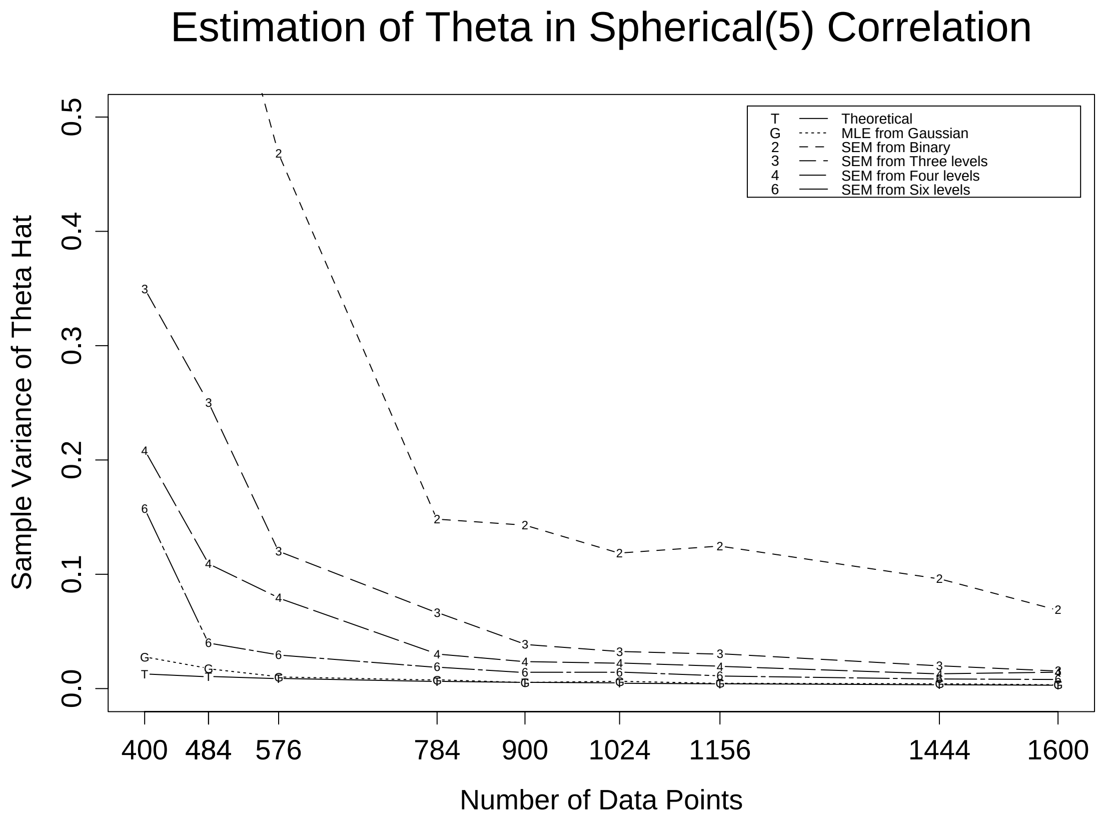

Figure 5.4: Variance of $\hat{\theta}$ when using MLE from the Gaussian data and SEM from the same data clipped at various number of thresholds.

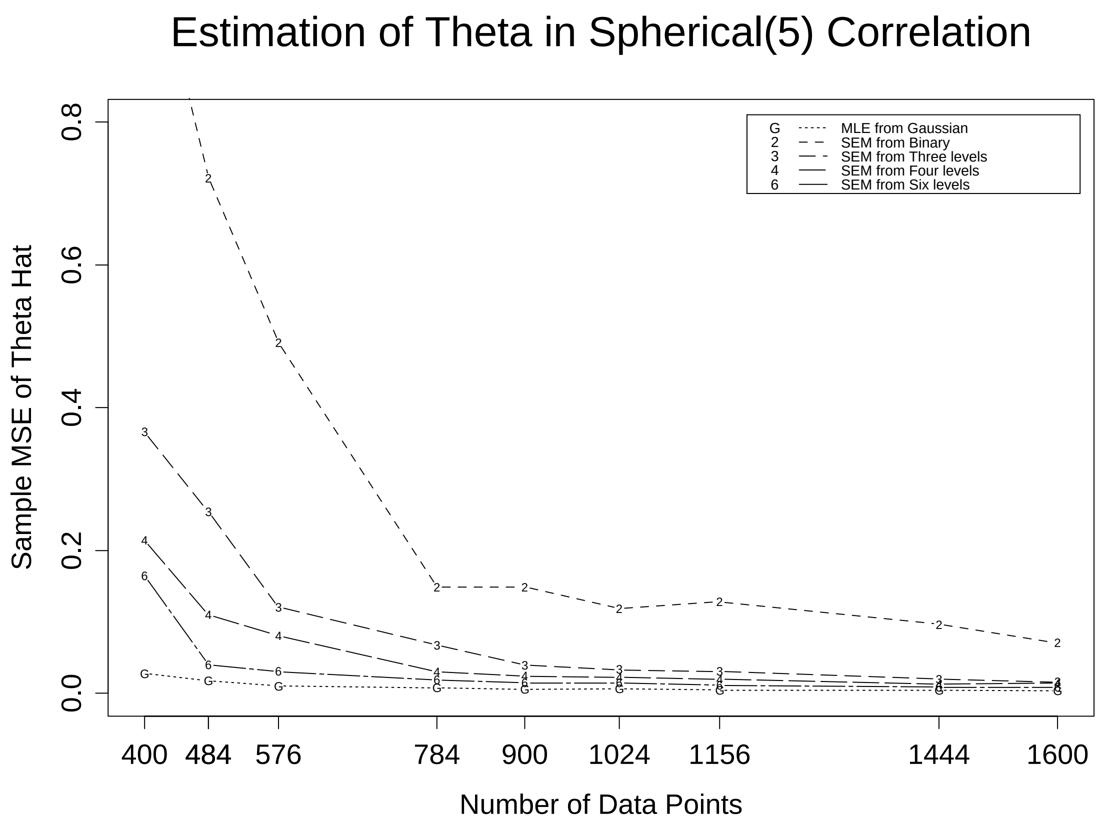

Figure 5.5: Mean squared error of $\hat{\theta}$ when using MLE from the Gaussian data and SEM from the same data clipped at various number of thresholds.

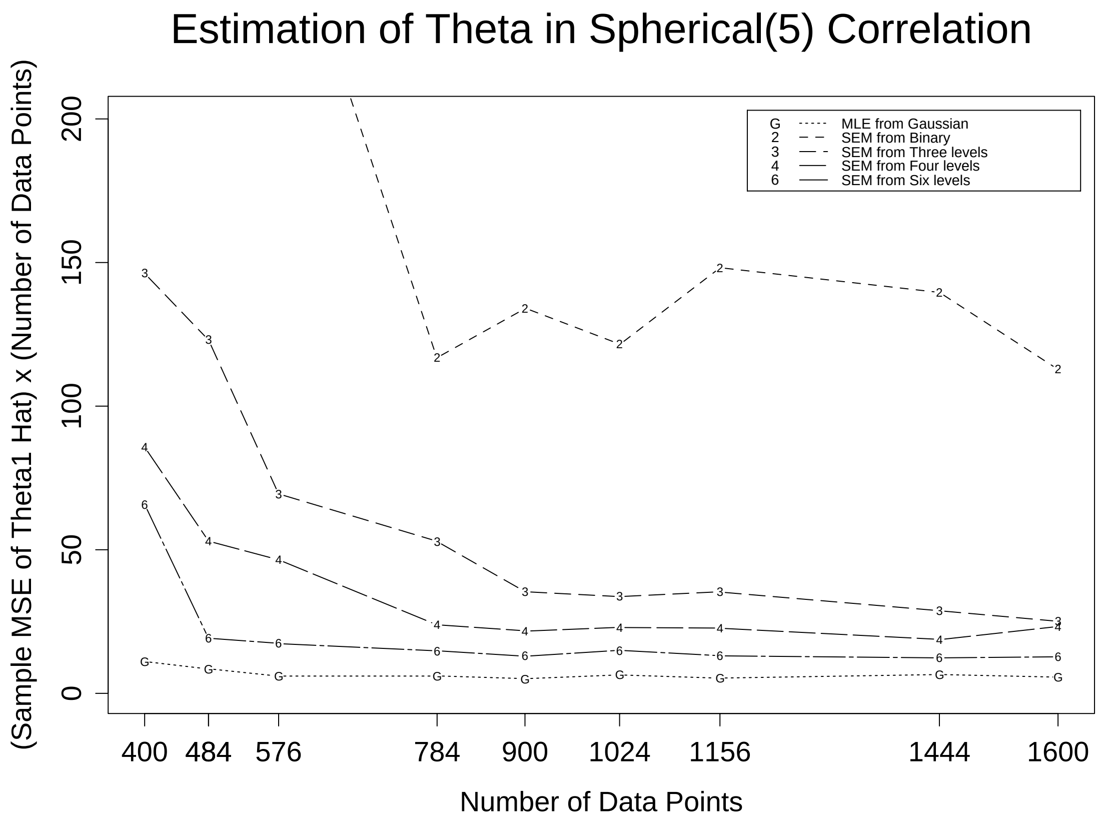

Figure 5.6: Mean squared error of $\hat{\theta}$ times the number of data points when using MLE from the Gaussian data and SEM from the same data clipped at various number of thresholds. Constant appearance of these graphs suggests C/n decay of the MSE.

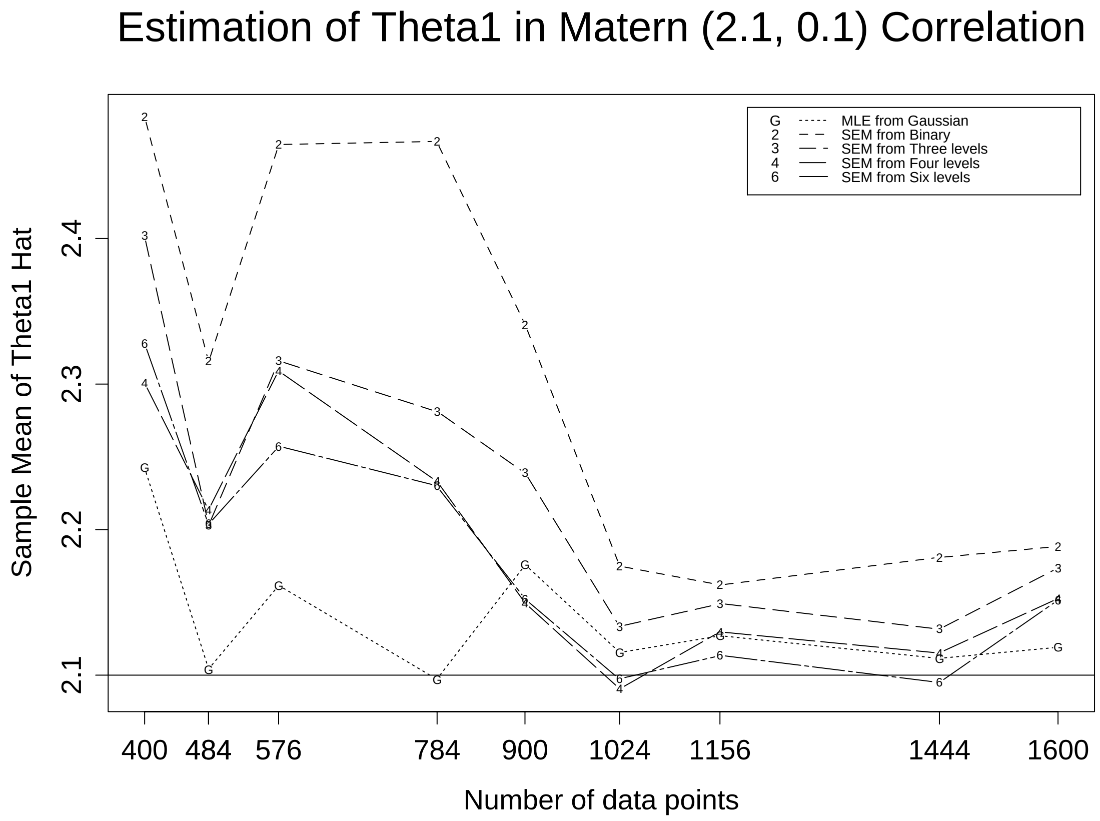

Figure 5.7: Mean of $\hat{\theta}$ when using MLE from the Gaussian data and SEM from the same data clipped at various number of thresholds.

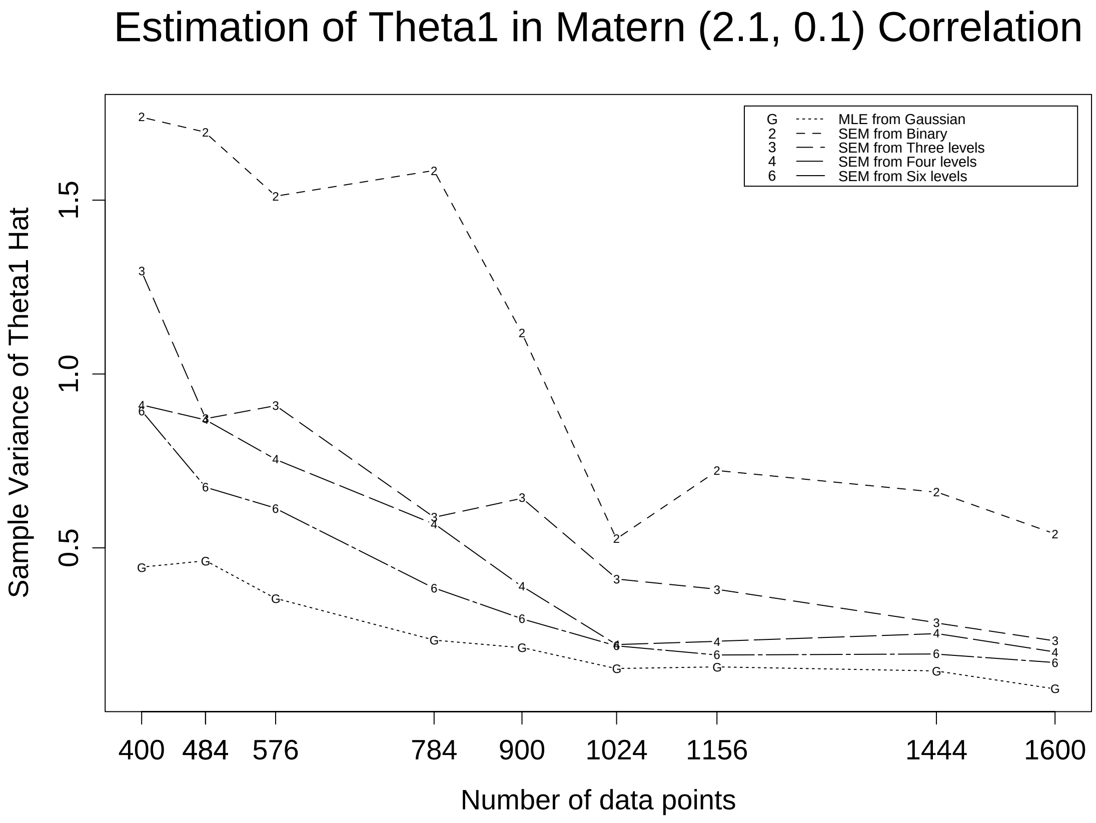

Figure 5.8: Variance of $\hat{\theta}$ when using MLE from the Gaussian data and SEM from the same data clipped at various number of thresholds.

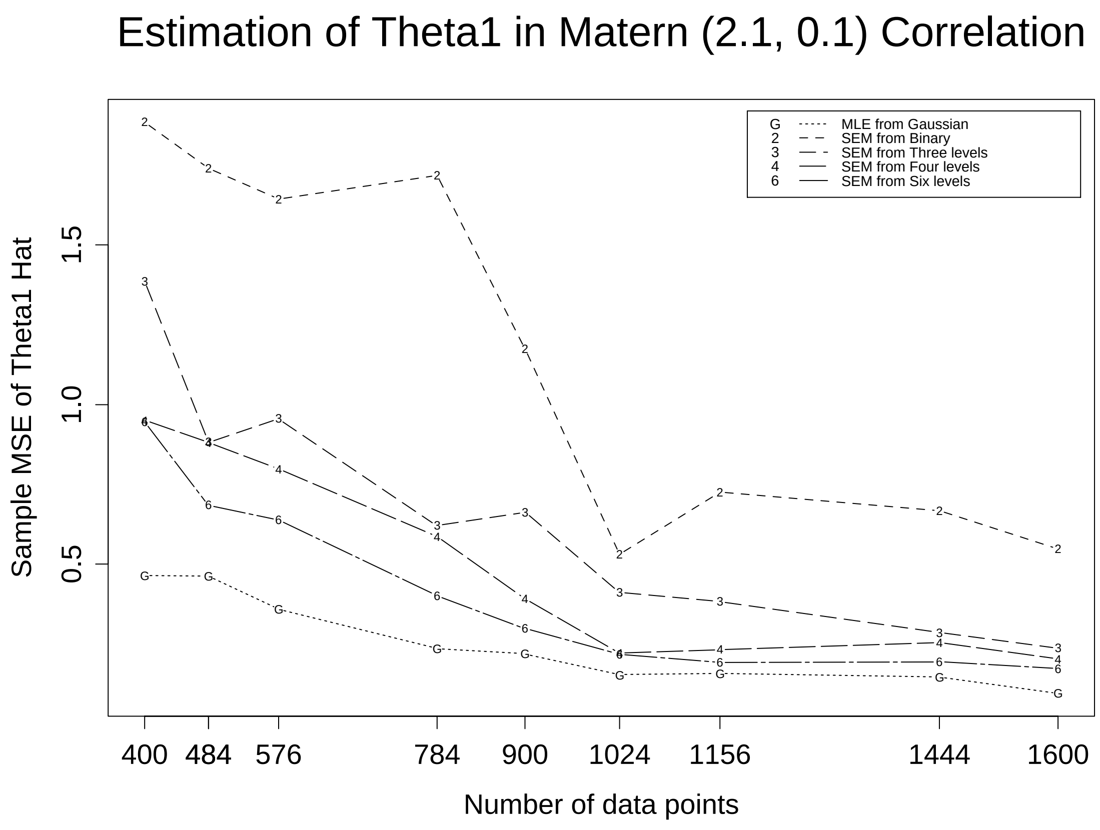

Figure 5.9: Mean squared error of $\hat{\theta}$ when using MLE from the Gaussian data and SEM from the same data clipped at various number of thresholds.

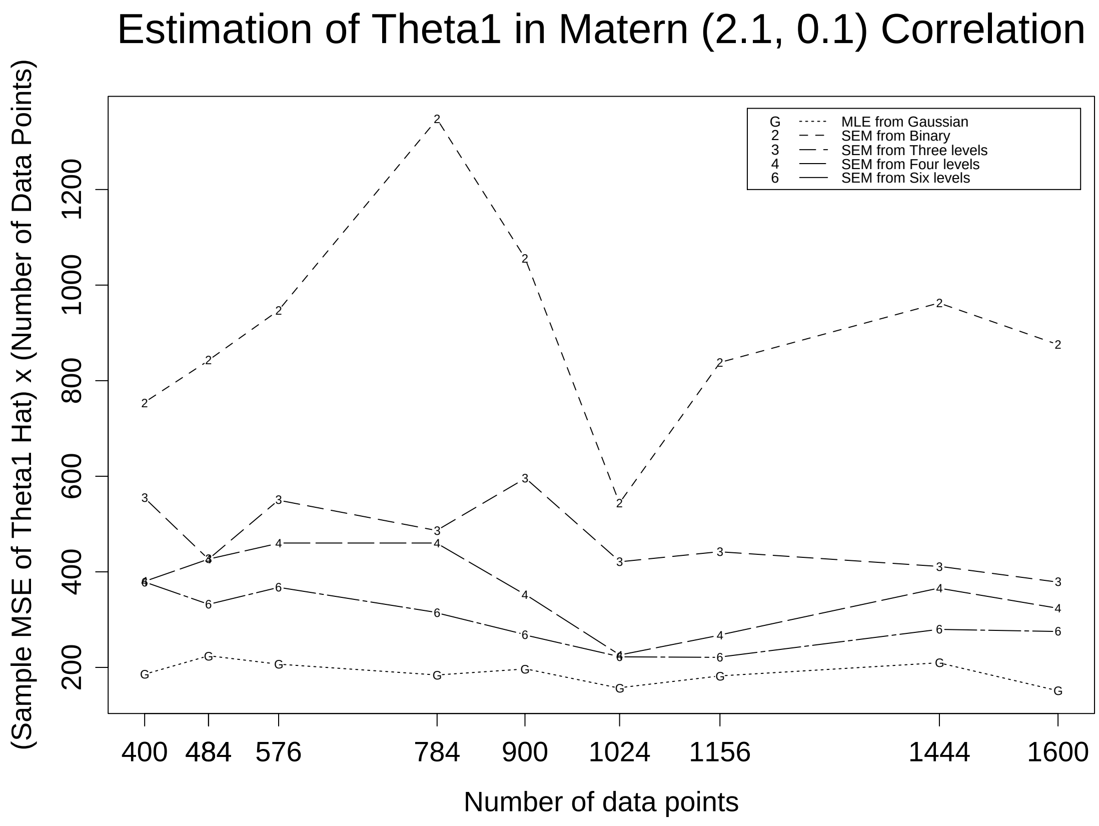

Figure 5.10: Mean squared error of $\hat{\theta}$ times the number of data points when using MLE from the Gaussian data and SEM from the same data clipped at various number of thresholds. Constant appearance of these graphs suggests C/n decay of the MSE.

Chapter 6

MAIN RESULTS AND FUTURE WORK

6.1 Main Results

This thesis described a powerful computational approach to spatial data models that are driven by stationary Gaussian fields over regular grids. The main idea of the approach is to take advantage of the covariance matrix structure after improving it (from Toeplitz to circulant) by artificially adding extra unobserved data points.

The first part (Chapters 2 and 3) provided fast algorithms for doing standard matrix operations with block circulant matrices (namely, computing the inverse, the determinant, the eigenvalues, and matrix times vector products). The Circulant Embedding method for exact unconditional generation of stationary Gaussian fields based on the "diagonalization by the FFT" technique is given.

The second part (Chapters 4 and 5) demonstrated how computationally tractable models can be built around stationary Gaussian fields using the methods of the first part. This was done in an example of the model for discrete spatial data, which extends the work of Nott and Wilson [47]. In this model, the discrete data $\mathbf{X}$ is treated as a quantization of the $\mathbf{Z}$-components of an unobserved Gaussian vector $\mathbf{W}$, where $\mathbf{W}$ embeds an unobserved Gaussian random field $\mathbf{Z}$.

The SEM algorithm for estimation of the vector parameter in the covariance function of $\mathbf{Z}$ was implemented. The idea of the implementation is to approximate the conditional expectation in the E-step of the EM algorithm with the mean of the corresponding conditional sample. To obtain such a sample, independent Gaussian vectors $\mathbf{W}^{(i)}$ need to be generated given $\mathbf{X}$, the observed quantization of the $\mathbf{Z}$-components of $\mathbf{W}^{(i)}$. This generation is performed approximately using the Gibbs sampler (an MCMC method).

The precision of the SEM estimation of the covariance parameter from the discrete data $\mathbf{X}$ was compared to that of the MLE estimation of the same parameter from the 'complete' Gaussian data $\mathbf{Z}$. The Fisher information matrix for Gaussian data was obtained analytically and compared to the empirical variance of the SEM estimators observed in the simulations.

In the simulations, we performed the following sequence of steps 250 times for various data sizes and various numbers of quantization levels.

- Generate a zero-mean Gaussian field $\mathbf{Z}$ with the covariance function $r(\cdot, \theta)$ unconditionally using the Circulant Embedding method;
- Quantize $\mathbf{Z}$ to obtain a discrete field $\mathbf{X}$;
- Estimate θ from $\mathbf{X}$ by the SEM algorithm:
  - Start with an arbitrary $\theta^{(0)}$;
  - Generate $\mathbf{W} \mid (\mathbf{X}, \theta^{(0)})$ using the Gibbs sampler;
  - Find $\theta^{(1)} = \arg\max_\theta \log L(\theta, \mathbf{W})$ by the numerical maximization;
  - $\ldots$ (repeat until convergence replacing 0 by $i-1$ and 1 by i).

Thus we obtained sample means and variances of the estimators of θ under various data sizes and various numbers of quantization levels.

6.2 Future Work

Speaking of the SEM algorithm implementation for the estimation of $\boldsymbol{\theta}$ in the discrete data model (Chapter 4), two issues have led to some criticism of the EM algorithm. The first concerns the provision of standard errors or the full covariance matrices of $\hat{\boldsymbol{\theta}}$, since the original EM does not automatically estimate these. The other common criticism that has been leveled against the EM algorithm is that its convergence can be quite slow. It would be interesting therefore to study recent techniques (McLachlan and Krishnan [44], Chapter 4) designed to alleviate these problems.

More generally, the computational methods described in this thesis can be immediately applied to any spatial model which uses stationary Gaussian fields in some form. We can modify such a model to be driven not by a usual Gaussian field (yielding block Toeplitz matrices), but by our encompassing Gaussian structures with block circulant matrices, perhaps by just ignoring the extra coordinates introduced during the embedding procedure, in the same way we did when we changed the unobserved data from $\mathbf{Z}$ to $\mathbf{W}$ in Section 4.4.

Even if we choose not to do so, we still have obtained a powerful simulating mechanism. We can now generate many Gaussian surfaces over large regions with various parameters and feed them into the model, to better study and understand it and to evaluate the precision of its results. In particular, it would be interesting to test the BTG model from de Oliveira [48] against, for example, trans-Gaussian kriging, using the simulated fields as input data.

BIBLIOGRAPHY

[1] Adler, R. J. (1981), *The Geometry of Random Fields*, New York: Wiley.

[2] Athans, M., and Schweppe, F. C. (1965), *Gradient matrices and matrix calculations*, Lincoln Lab. Tech. Note 1965-53, MIT, Lexington, Mass.

[3] Bhat, B. R. (1974), "On the method of maximum likelihood for dependent observations," *Journal of Royal Statistical Society*, B, 36, 48-53.

[4] Billingsley, P. (1961), *Statistical Inference for Markov Processes*, University of Chicago Press.

[5] Bindel, D. (1997), *The `btg` program*, `http://www.math.umd.edu/~bnk/btg_page.html`.

[6] Borgman, L., Taheri, M. and Hagan, R. (1984), "Three-dimensional frequency-domain simulations of geological variables," in *Geostatistics for Natural Resource Characterization*, edited by G.Verly, M.David, A.G.Journel, and A.Marechal, 517-541, D. Reidel, Norwell, Mass.

[7] Breiman, L. (1968), *Probability,* Addison-Wesley, Reading, Mass.

[8] Brooks, S. P., and Roberts, G. O. (1995), "Review of Convergence Diagnostics," *Technical Report*, Cambridge: University of Cambridge.

[9] Casella, G., and Berger, R. L. (1990), *Statistical Inference*, Duxbury Press, Calif.

[10] Casella, G., and George, E. I. (1992), "Explaining the Gibbs sampler," *The American Statistician*, 46, 167-174.

[11] Chilès, J.-P., and Delfiner, P. (1999), *Geostatistics. Modeling Spatial Uncertainty*, New York: Wiley.

[12] Cody, W. J. (1988), *SPECFUN 2.5. A collection of transportable Fortran programs for special functions and accompanying test programs*, `http://www.netlib.org/specfun/`.

[13] Cooley, J. W., and Tukey, J. W. (1965), "An algorithm for the machine computation of the complex Fourier series," *Mathematics of Computation*, 19, 297-301.

[14] Cowles, M. K., and Carlin, B. P. (1996), "Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review," *Journal of the American Statistical Association*, 91, 883-904.

[15] Cressie, N. A. C. (1993), *Statistics for Spatial Data*, New York: Wiley.

[16] Crowder, M. J. (1976), "Maximum likelihood estimation for dependent observations," *Journal of the Royal Statistical Society*, B, 38, 45-53.

[17] Davies, R. B., and Harte, D. S. (1987), "Tests for Hurst Effect," *Biometrika*, 74, 95-101.

[18] Dembo, A., Mallows, C. L., Shepp, L. A. (1989), "Embedding nonnnegative definite Toeplitz matrices in nonnegative definite circulant matrices, with

application to covariance estimation," *IEEE Transactions on Information Theory*, 35, 1206-1212.

[19] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977), "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society*, Series B, 39, 1-38.

[20] Devroye, L. (1986), *Non-Uniform Random Variate Generation*, New-York: Springer-Verlag.

[21] Diebolt, J., and Ip, E. H. S. (1996), "Stochastic EM: method and application," in *Markov Chain Monte Carlo in Practice*, eds. W. R. Gilks, S. Richardson, and D. I. Spiegelhalter, London: Chapman and Hall, 259-273.

[22] Dietrich, C., and Newsam, G. (1993), "A Fast and Exact Method for Multidimensional Gaussian Stochastic Simulations," *Water Resources Research*, 29, 2861-2869.

[23] Dietrich, C., and Newsam, G. (1997), "Fast and exact simulation of stationary Gaussian processes through circulant embedding of the covariance matrix," *SIAM Journal of Scientific Computation*, 18, 4, 1088-1107.

[24] Forsythe, G. E., Malcolm, M. A., and Moler, C. B. (1977), *Computer Methods for Mathematical Computations*, Englewood Cliffs, N.J.: Prentice-Hall.

[25] Freulon, X., and de Fouquet, C. (1993), "Conditioning a Gaussian model with inequalities," in A. Soares, (Ed.), *Geostatistics Tróia* '92, Vol. 1, Kluwer Academic Publishers, Dordrecht, 61-72.

[26] Frigo, M., and Johnson, S. G. (1997), "FFTW: An Adaptive Software Architecture for the FFT," *23rd International Conference on*

Acoustics, Speech, and Signal Processing (ICASSP '98), 3, 1381, `http://theory.lcs.mit.edu/~fftw`.

[27] Gilks, W. R., Richardson, S., and Spiegelhalter, D. J. (Eds.) (1996), *Markov Chain Monte Carlo in Practice*, London: Chapman and Hall.

[28] Golub, G. H., and Van Loan, C. F. (1996), *Matrix Computations*, 3rd ed., Baltimore: Johns Hopkins University Press.

[29] Hastings, W. K. (1970), "Monte Carlo sampling methods using Markov chains and their applications." *Biometrika*, 57, 97-109.

[30] Hjort, N. L, and Omre, H. (1994), "Topics in Spatial Prediction" (with discussion), *Scandinavian Journal of Statistics*, 21, 189-357.

[31] Johnson, M. (1994), *Nonlinear optimization using the algorithm of Hooke and Jeeves*, 1994, `http://www.netlib.org/opt/hooke.c`.

[32] Kaupe Jr., A. F. (1963), "Algorithm 178: Direct Search," *Communications of the ACM*, 6, p.313.

[33] Kass, R. E., Carlin, B. P., Gelman, A., and Neal, R. M. (1998), "Markov Chain Monte Carlo in Practice: A Roundtable Discussion," *The American Statistician*, 1998, 52, 93-100.

[34] Kedem, B. (1980), *Binary Time Series*, New York: M. Dekker.

[35] Kedem, B., Pfeiffer, R., and Short, D. A. (1997), "Variability of Space-Time Mean Rain Rate," *Journal of Applied Meteorology*, 36, 443-451.

[36] Keselyov, O. (1991), *Function FMINBR - one-dimensional search for a function minimum over the given range*, `http://www.netlib.org/c/serv.shar`.

[37] Kitanidis, P. K. (1983), "Statistical Estimation of Polynomial Generalized Covariance Functions and Hydrologic Applications," *Water Resourses Research*, 19, 909-921.

[38] Kitanidis, P. K., and Lane, R. W. (1985), "Maximum Likelihood Parameter Estimation of Hydrologic Spatial Processes by the Gauss-Newton Method," *Journal of Hydrology*, 79, 53-71.

[39] Lindstrom, M. J., and Bates, D. M. (1988), "Newton-Raphson and EM algorithms for linear mixed-effects models for repeated-measures data," *Journal of the Americal Statistical Association*, 83, 1014-1022.

[40] Mantoglou, A., and Wilson, J. L. (1982), "The turning bands methods for simulation of random fields using line generation by a spectral method," *Water Resourses Research*, 18, 1379-1394.

[41] Mardia, K. V., and Marshall, R. J. (1984), "Maximum likelihood estimation of models for residual covariance in spatial regression," *Biometrika*, 71, 135-146.

[42] Mardia, K. V., and Watkins, A. J. (1989), "On multimodality of the likelihood in the spatial linear model," *Biometrika*, 76, 289-295.

[43] Matérn, B. (1986), *Spatial Variation* (2nd. ed.), Lecture Notes in Statistics, 36, Berlin: Springer-Verlag.

[44] McLachlan, G., and Krishnan, T. (1997), *The EM Algorithm and Extensions*, New York: Wiley.

[45] Mejia, J. M. and Rodrigez-Iturbe, I. (1974), "On the synthesis of random field sampling from the spectrum: An application to the generation of hydrological spatial process," *Water Resources Research,* 10, 705-711.

[46] Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller A. H., and Teller, E. (1953), "Equations of state calculations by fast computing machine." *Journal of Chemical Physics*, 21, 1087-1091.

[47] Nott, D. J., and Wilson, R. J. (1997), "Parameter estimation for excursion set texture models," *Signal Processing*, 63, 199-210.

[48] de Oliveira, V., Kedem, B., and Short, D. (1997), "Bayesian Prediction of Transformed Gaussian Random Fields," *Journal of the American Statistical Association*, 92, 440, 1422-1433.

[49] de Oliveira Martinez, V. (1997), *Prediction in some classes of non-Gaussian random fields*, Ph.D. dissertation, University of Maryland.

[50] Perlman, G. (1987), *Compute approximations to normal z distribution probabilities*, `http://www.netlib.org/a/perlman`.

[51] Press, W. H., Teukolsky, S. A., Vetterling, W. T. and Flannery, B. P. (1992), *Numerical Recipes in C. The Art of Scientific Computing. Second Edition*, Cambridge University Press, `http://beta.ul.cs.cmu.edu/webRoot/Books/Numerical_Recipes/bookcpdf.html`.

[52] Priestley, M. B. (1981), *Spectral Analysis and Time Series*, vol.1: Univariate Series, London; New York: Academic Press.

[53] Raftery, A. E., and Lewis, S. M. (1996), "Implementing MCMC," in *Markov Chain Monte Carlo in Practice*, eds. W. R. Gilks, S. Richardson, and D. I. Spiegelhalter, London: Chapman and Hall, 259-273.

[54] Ripley, B. D. (1987), *Stochastic Simulation*, New York: Wiley.

[55] Shinozuka, M., and Jan, C. M. (1972), "Digital simulation of random processes and its applications," *Journal of Sound and Vibration*, 25(1), 111-128.

[56] Sjöström, E. (1996), *Singular Value Computations for Toeplitz Matrices*, Linköping Studies in Science and Technology, Thesis No. 554, `http://www.mai.liu.se/~evlun/pub/lic/lic.html`.

[57] Stoer, J., and Bulirsch, R. (1993), *Introduction to Numerical Analysis*, Second Edition, New York: Springer-Verlag.

[58] Sweeting, T. J. (1980), "Uniform asymptotic normality of the maximum likelihood estimator," *Annals of Statistics*, 8(6), 1375-1381.

[59] Tierney, L. (1994), "Markov chains for exploring posterior distributions (with discussion)," *Annals of Statistics*, 22, 1701-1762.

[60] Tompson, A. F. B., Ababou, R., and Gelhar, L. W. (1989), "Implementation of the three-dimensional turning bands field generator," *Water Resourses Research*, 25(10), 2227-2243.

[61] Trapp, G. E. (1973), "Inverses of Circulant Matrices and Block Circulant Matrices," *Kyungpook Mathematical Journal*, 13, 1, 11-20.

[62] Warnes, J. J., and Ripley, B. D. (1987), "Problems with the likelihood estimation of covariance functions of spatial Gaussian processes," *Biometrika*, 74(3), 640-642.

[63] Wood, A. T. A., and Chan, G. (1994), "Simulation of stationary Gaussian Processes in $[0,1]^d$," *Journal of Computational and Graphical Statistics*, 3(4), 409-432.

[64] Wu, C. F. J. (1983), "On the convergence properties of the EM algorithm," *Annals of Statistics*, 11, 95-103.

[65] Yaglom, A. M., (1987), *Correlation Theory of Stationary and Related Random Functions I. Basic Results*, New York: Springer-Verlag.

[66] Zimmerman, D. L. (1989), "Computationally exploitable structure of covariance matrices and generalized covariance matrices in spatial models," *Journal of Statistical Computation and Simulation*, 32, 1-15.