MASTER'S THESIS

Comparison of Three Methods of Spatial Prediction

by Alexandra Kozintseva
Advisor: Benjamin Kedem

M.S. 99-12

ISR develops, applies and teaches advanced methodologies of design and analysis to solve complex, hierarchical, heterogeneous and dynamic problems of engineering technology and systems for industry and government.

ISR is a permanent institute of the University of Maryland, within the Glenn L. Martin Institute of Technology/A. James Clark School of Engineering. It is a National Science Foundation Engineering Research Center.

Web site http://www.isr.umd.edu

ABSTRACT

Title of Thesis:	Comparison of Three Methods of Spatial Prediction
Degree candidate:	Alexandra Kozintseva
Degree and year:	Master of Arts, 1999
Thesis directed by:	Professor Benjamin Kedem Department of Mathematics

Three methods for spatial prediction in Gaussian and transformed Gaussian random fields are described and compared.

The first two methods are ordinary kriging and trans-Gaussian kriging.

The third method is the Bayesian Transformed Gaussian model (BTG), which provides an alternative to trans-Gaussian kriging by taking into account the uncertainty about the exact parameter in the 'normalizing transformation'.

All three methods were applied to the simulated data sets for each of four correlation families (exponential, rational quadratic, spherical, and Matérn), and to actual rainfall intensity data sets. The normalizing transformation was selected from the family of Box-Cox transformations.

Cross validation on the simulated data shows that all three methods are close in terms of the mean squared error (MSE) and that BTG provides more realistic prediction intervals. The analysis of the rainfall data in terms of cross-validation shows that kriging and BTG are comparable.

Comparison of Three Methods of Spatial Prediction

by

Alexandra Kozintseva

Thesis 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
Master of Arts
1999

Advisory Committee:

Professor Benjamin Kedem, Chairman/Advisor
Professor Brian Hunt
Professor Paul Smith

© Copyright by

Alexandra Kozintseva

1999

DEDICATION

To my parents and Boris

ACKNOWLEDGEMENTS

I would like to thank my advisor Professor Benjamin Kedem. I was very lucky to have him as an advisor during my first research experience. His enthusiasm and support helped a lot during these past three years.

I want to thank Victor de Oliveira for help during this project and David Bindel for writing the BTG program.

I also want to thank Dr. David Wolff of NASA Goddard Space Flight Center for preparing the data sets.

Financial support from NASA Grant NAG52783 is also appreciated.

Finally, I want to thank Boris Kozintsev. His love, patience, and programming advice made it possible to finish this study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 The Problem

The need to obtain an accurate prediction from observed data can be found in all scientific disciplines. In many cases a quantity of interest, z, varies over a domain in space according to an unknown function $z : D \subset \mathbb{R}^d \to \mathbb{R}$, in a very complicated way. In geostatistical literature this function is called a *regionalized variable*, to emphasize the continuous nature of the spatial set D. In many cases it is too expensive, too time consuming or even impossible to study $z(.)$ by extensive experimentation. In these situations z is measured at a small set of locations, and inference about z for many other unobserved (and often unobservable) locations in the domain of interest is based on a proposed mathematical model for the function $z(.)$. A good example is rainfall measured at a few locations in a given region. Rainfall amounts in other locations are "predicted" conditional on the observed amounts.

One possible approach to this problem is to treat the unknown function $z(.)$ as a realization of a *random field* $\{Z(\mathbf{s}), \mathbf{s} \in D\}$. Then the observed data are some part of a realization of this random field and we want to predict an unobserved

part of the same realization. To do so we can use the available data and the known (or assumed) dependency structure of the random field. This approach provides a mathematical framework for efficient predictions at unobserved locations and for computing uncertainty measures associated with these predictions. Inference is usually summarized, for an unobserved location $\mathbf{s}_0 \in D$, by the pair $\left(\widehat{Z}(\mathbf{s}_0), \widehat{\sigma}(\mathbf{s}_0)\right)$, where $\widehat{Z}(\mathbf{s}_0)$ is the predictor for $Z(\mathbf{s}_0)$, the value of z at location $\mathbf{s}_0$, and $\widehat{\sigma}(\mathbf{s}_0)$ is a measure of prediction uncertainty associated with $\widehat{Z}(\mathbf{s}_0)$.

In this work we will consider three ways to do the prediction: ordinary kriging, which is the most popular stochastic method (Ripley, 1981), trans-Gaussian kriging (Cressie, 1993), and the Bayesian Transformed Gaussian model, BTG, (De Oliveira, Kedem, Short, 1997). We will compare them using simulated and real data by means of cross-validation, and mean squared prediction error.

1.2 Thesis Synopsis

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

Chapter 2 describes the algorithms for ordinary and trans-Gaussian kriging, defines the Box-Cox transformation, and gives implementations of these algorithms in terms of S-Plus.

Chapter 3 studies transformed Gaussian random fields, where the transformation is known to belong to a parametric family of monotone transformations. A Gaussian random field is a special case of this model.

Chapter 4 describes a method to generate Gaussian random fields with a known correlation structure.

Chapter 5 provides the numerical results of the comparison of the three methods using generated and real data.

Chapter 2

Kriging

2.1 Ordinary Kriging

In this method we want to predict $Z(\mathbf{s}_0)$ using the best linear unbiased predictor (BLUP) based on the observed data. BLUP depends only on the second-order properties of the random field and no full distributional assumptions are made. We are given n observations $Z(\mathbf{s}_i)$ and we wish to map the process $Z(\mathbf{s})$ within a region D. We assume that all the sample points $\mathbf{s}_i$ are within a two-dimensional domain D.

The data

$$\mathbf{Z} \equiv (Z(\mathbf{s}_1), \ldots, Z(\mathbf{s}_n))^T$$

are observed at known spatial locations $\{\mathbf{s}_1, \ldots, \mathbf{s}_n\} \subset D$.

Definition:

A random function $\mathbf{Z}(.)$ satisfying

$$E(\mathbf{Z}(\mathbf{s})) = \mu, \text{ for all } \mathbf{s} \in D$$

and

$$\text{Cov}(\mathbf{Z}(\mathbf{s}_1), \mathbf{Z}(\mathbf{s}_2)) = C(\mathbf{s}_1 - \mathbf{s}_2), \text{ for all } \mathbf{s}_1, \mathbf{s}_2 \in D$$

is called *second-order stationary*. The function $C(.)$ is called the *covariogram* or stationary covariance function.

The correlation functions (normalized covariance functions) that we are going to use are not just stationary, but also isotropic - they depend only on the distance between the points, but not on the direction.

The value of the correlation function at zero is always equal to one, but since the variance of the field does not have to be one we need to define the *precision* of the random field τ. A parametric relationship between the covariance and correlation functions is

$$\mathrm{Cov}(\mathbf{Z}(\mathbf{s}_i), \mathbf{Z}(\mathbf{s}_j)) = C(\mathbf{s}_i - \mathbf{s}_j) = \frac{1}{\tau} r_{\boldsymbol{\theta}}(\mathbf{s}_i - \mathbf{s}_j),$$

and

$$\mathrm{Cov}(\mathbf{Z}(\mathbf{s}_i), \mathbf{Z}(\mathbf{s}_i)) = C(\mathbf{0}) = \mathrm{Var}(Z) = \frac{1}{\tau} r_{\boldsymbol{\theta}}(0) = \frac{1}{\tau} 1,$$

where $r_{\boldsymbol{\theta}}(\mathbf{s}_i - \mathbf{s}_j)$ is the correlation function and $\boldsymbol{\theta}$ is the parameter. We denote by $r_{\boldsymbol{\theta}}(l)$ an isotropic correlation function with parameter $\boldsymbol{\theta}$, and l is the distance between the points where the covariance is computed.

These are some of the commonly used correlation function families.

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 $\theta > 0$.

The *ordinary kriging* predictor satisfies the following two assumptions.

1. Model Assumption:

the field Z is second order stationary with the unknown mean μ,

$$Z(\mathbf{s}) = \mu + \delta(\mathbf{s}), \quad \mathbf{s} \in D, \quad \mu \in \mathbb{R}, \tag{2.1}$$

where $\delta(.)$ is a zero-mean, second-order stationary process with covariogram $C(\mathbf{h}), \mathbf{h} \in \mathbb{R}^2$.

2. Predictor Assumption: the predictor $p(\mathbf{Z}; \mathbf{s}_0)$ is linear and satisfies

$$p(\mathbf{Z}; \mathbf{s}_0) = \sum_{i=1}^{n} \lambda_i Z(\mathbf{s}_i), \qquad \text{where} \sum_{i=1}^{n} \lambda_i = 1. \tag{2.2}$$

The condition that the coefficients of the linear predictor sum to 1 guarantees uniform unbiasedness:

$$E(p(\mathbf{Z}; \mathbf{s}_0)) = E\left(\sum_{i=1}^{n} \lambda_i Z(\mathbf{s}_i)\right)$$

$$= E(\mathbf{Z}(\mathbf{s}_0)) \sum_{i=1}^{n} \lambda_i$$

$$= \mu.$$

Of all the predictors satisfying the model and prediction assumptions, the *ordinary kriging predictor* is defined as an optimal one, where the word "optimal" refers to the squared-error loss. Therefore the goal is to minimize the mean-squared prediction error

$$\sigma_e^2(\mathbf{s}_0) \equiv E\big(Z(\mathbf{s}_0) - p(\mathbf{Z}; \mathbf{s}_0)\big)^2$$

with respect to the predictor coefficients.

To find the ordinary kriging predictor we need to minimize the function f,

$$f(\lambda_1, \ldots, \lambda_n, m) = E\big(Z(\mathbf{s}_0) - p(\mathbf{Z}; \mathbf{s}_0)\big)^2 - 2m \left(\sum_{i=1}^{n} \lambda_i - 1 \right) \tag{2.3}$$

with respect to $\lambda_1, \ldots, \lambda_n$ and m (the parameter m is a Lagrange multiplier that guarantees $\sum_{i=1}^{n} \lambda_i = 1$). For our model (2.1) and predictor (2.2) the expression (2.3) becomes

$$\begin{aligned}
&f(\lambda_1, \ldots, \lambda_n, m) \\
&\quad = E\left(Z(\mathbf{s}_0) - \sum_{i=1}^{n} \lambda_i Z(\mathbf{s}_i) \right)^2 - 2m \left(\sum_{i=1}^{n} \lambda_i - 1 \right) \\
&\quad = E\big(Z(\mathbf{s}_0)^2\big) - 2E\left(Z(\mathbf{s}_0) \sum_{i=1}^{n} \lambda_i Z(\mathbf{s}_i) \right) + E\left(\left(\sum_{i=1}^{n} \lambda_i Z(\mathbf{s}_i) \right)^2 \right) \\
&\qquad - 2m \left(\sum_{i=1}^{n} \lambda_i - 1 \right) \\
&\quad = C(\mathbf{0}) + \mu^2 - 2 \sum_{i=1}^{n} \lambda_i E\big(Z(\mathbf{s}_0) Z(\mathbf{s}_i)\big) \\
&\qquad + \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j E\big(Z(\mathbf{s}_i) Z(\mathbf{s}_j)\big) - 2m \left(\sum_{i=1}^{n} \lambda_i - 1 \right)
\end{aligned}$$

$$
\begin{aligned}
&= C(\mathbf{0}) + \mu^2 - 2\sum_{i=1}^{n} \lambda_i \big(C(\mathbf{s}_0 - \mathbf{s}_i) + \mu^2\big) \\
&\quad + \sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i\lambda_j \big(C(\mathbf{s}_i - \mathbf{s}_j) + \mu^2\big) - 2m\left(\sum_{i=1}^{n}\lambda_i - 1\right) \\
&= C(\mathbf{0}) + \sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i\lambda_j C(\mathbf{s}_i - \mathbf{s}_j) \qquad (2.4) \\
&\quad - 2\sum_{i=1}^{n} \lambda_i C(\mathbf{s}_0 - \mathbf{s}_i) - 2m\left(\sum_{i=1}^{n}\lambda_i - 1\right), \text{ since} \sum\sum \lambda_i\lambda_j = 1 \\
&= C(\mathbf{0}) + \sum_{i=j} \lambda_i\lambda_j C(\mathbf{s}_i - \mathbf{s}_j) + \sum_{i\neq j} \lambda_i\lambda_j C(\mathbf{s}_i - \mathbf{s}_j) \\
&\quad - 2\sum_{i=1}^{n} \lambda_i C(\mathbf{s}_0 - \mathbf{s}_i) - 2m\left(\sum_{i=1}^{n}\lambda_i - 1\right) \\
&= C(\mathbf{0}) + C(\mathbf{0})\sum_{i=1}^{n}\lambda_i^2 + 2\sum_{i<j} \lambda_i\lambda_j C(\mathbf{s}_i - \mathbf{s}_j) - 2\sum_{i=1}^{n} \lambda_i C(\mathbf{s}_0 - \mathbf{s}_i) \\
&\quad - 2m\left(\sum_{i=1}^{n}\lambda_i - 1\right).
\end{aligned}
$$

We need to minimize f with respect to $\lambda_1, \ldots, \lambda_n$ and m. Note that f is a second degree polynomial in $\lambda_1, \ldots, \lambda_n$ and m.

Differentiating, we get

$$
\begin{aligned}
\frac{\partial f}{\partial \lambda_1} &= 2\lambda_1 C(\mathbf{0}) + 2\sum_{1<j} \lambda_j C(\mathbf{s}_1 - \mathbf{s}_j) - 2C(\mathbf{s}_0 - \mathbf{s}_1) - 2m \\
&= 2\sum_{j=1}^{n} \lambda_j C(\mathbf{s}_1 - \mathbf{s}_j) - 2C(\mathbf{s}_0 - \mathbf{s}_1) - 2m \\
&\vdots \\
\frac{\partial f}{\partial \lambda_n} &= 2\lambda_n C(\mathbf{0}) + 2\sum_{j\neq n} \lambda_j C(\mathbf{s}_n - \mathbf{s}_j) - 2C(\mathbf{s}_0 - \mathbf{s}_n) - 2m \\
&= 2\sum_{j=1}^{n} \lambda_j C(\mathbf{s}_n - \mathbf{s}_j) - 2C(\mathbf{s}_0 - \mathbf{s}_n) - 2m \\
\frac{\partial f}{\partial m} &= -2\left(\sum_{i=1}^{n}\lambda_i - 1\right)
\end{aligned}
$$

Now we need to solve the linear system:

$$\begin{cases} \sum_{j=1}^{n} \lambda_j C(\mathbf{s}_1 - \mathbf{s}_j) = C(\mathbf{s}_0 - \mathbf{s}_1) + m \\ \dots \\ \sum_{j=1}^{n} \lambda_j C(\mathbf{s}_n - \mathbf{s}_j) = C(\mathbf{s}_0 - \mathbf{s}_n) + m \\ \sum_{i=1}^{n} \lambda_i = 1 \end{cases}$$

Denote

$$\mathbf{1} = (1, 1, \dots, 1)^T_{1\times n},$$

$$\mathbf{c} = \big(C(\mathbf{s}_0 - \mathbf{s}_1), \dots, C(\mathbf{s}_0 - \mathbf{s}_n)\big)^T,$$

$\mathbf{C}$ is an $n \times n$ matrix with $C_{ij} = C(\mathbf{s}_i - \mathbf{s}_j)$,

$$\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \dots, \lambda_n)^T$$

$$\mathbf{m} = m\mathbf{1}.$$

Then we can rewrite the system as

$$\begin{cases} \mathbf{C}\boldsymbol{\lambda} = \mathbf{c} + \mathbf{m} \\ \sum_{i=1}^{n} \lambda_i = 1. \end{cases} \tag{2.5}$$

The first equation gives

$$\boldsymbol{\lambda} = \mathbf{C}^{-1}(\mathbf{c} + \mathbf{m}).$$

Plugging this into the second equation we get

$$\sum_{i=1}^{n} \lambda_i = \mathbf{1}^T\boldsymbol{\lambda} = \mathbf{1}^T\mathbf{C}^{-1}(\mathbf{c} + \mathbf{m}) = \mathbf{1}^T\mathbf{C}^{-1}\mathbf{c} + \mathbf{1}^T\mathbf{C}^{-1}\mathbf{m} = 1.$$

Therefore,

$$m = \frac{1 - \mathbf{1}^T\mathbf{C}^{-1}\mathbf{c}}{\mathbf{1}^T\mathbf{C}^{-1}\mathbf{1}}$$

$$= \frac{1 - \sum_i (\mathbf{C}^{-1}\mathbf{c})_i}{\sum_i \sum_j (\mathbf{C}^{-1})_{ij}}. \tag{2.6}$$

So the λ that minimizes f is given by

$$\widehat{\boldsymbol{\lambda}} = \mathbf{C}^{-1}\left(\mathbf{c} + \frac{1 - \sum_i (\mathbf{C}^{-1}\mathbf{c})_i}{\sum_i \sum_j (\mathbf{C}^{-1})_{ij}}\mathbf{1}\right), \tag{2.7}$$

and the ordinary kriging predictor is

$$\widehat{p}(\mathbf{Z}; \mathbf{s}_0) = \widehat{\boldsymbol{\lambda}}^T \mathbf{Z}.$$

To simplify the notation we write $\boldsymbol{\lambda}$ for $\widehat{\boldsymbol{\lambda}}$.

The minimized mean-squared prediction error is called the *kriging variance* and is denoted by $\sigma_k^2(\mathbf{s}_0)$. For the ordinary kriging predictor $\widehat{p}(\mathbf{Z}; \mathbf{s}_0)$, the kriging variance is

$$\begin{aligned}
\sigma_k^2(\mathbf{s}_0) &= E\big(Z(\mathbf{s}_0) - \widehat{p}(\mathbf{Z}; \mathbf{s}_0)\big)^2 \\
&= C(\mathbf{0}) + \sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j C(\mathbf{s}_i - \mathbf{s}_j) - 2\sum_{i=1}^n \lambda_i C(\mathbf{s}_0 - \mathbf{s}_i) \quad \text{(from (2.4))} \\
&= C(\mathbf{0}) + \boldsymbol{\lambda}^T \mathbf{C} \boldsymbol{\lambda} - 2\boldsymbol{\lambda}^T \mathbf{c} \\
&= C(\mathbf{0}) + \boldsymbol{\lambda}^T (\mathbf{c} + \mathbf{m}) - 2\boldsymbol{\lambda}^T \mathbf{c} \qquad \text{(from (2.5))} \\
&= C(\mathbf{0}) - \boldsymbol{\lambda}^T \mathbf{c} + \boldsymbol{\lambda}^T \mathbf{m} \\
&= C(\mathbf{0}) - \boldsymbol{\lambda}^T \mathbf{c} + m.
\end{aligned}$$

So,

$$\sigma_k^2(\mathbf{s}_0) = C(\mathbf{0}) - \boldsymbol{\lambda}^T \mathbf{c} + m, \tag{2.8}$$

where m and $\boldsymbol{\lambda}$ are defined in (2.6) and (2.7).

Under the assumption that $Z(.)$ is Gaussian,

$$A \equiv [\hat{Z}(\mathbf{s}_0) - 1.96\sigma_k(\mathbf{s}_0), \hat{Z}(\mathbf{s}_0) + 1.96\sigma_k(\mathbf{s}_0)],$$

where $\hat{Z}(\mathbf{s}_0) = \hat{p}(\mathbf{Z}, \mathbf{s}_0)$ is the 95% prediction interval for $Z(\mathbf{s}_0)$, which means that

$$P(Z(\mathbf{s}_0) \in A) = .95.$$

This is not necessarily true for non-Gaussian data.

Therefore, the algorithm for ordinary kriging is as follows.

Given the values $Z(\mathbf{s}_1), \ldots, Z(\mathbf{s}_n)$ at points $\mathbf{s}_1, \ldots, \mathbf{s}_n$, the covariogram function $C(.)$ and the location of interest $\mathbf{s}_0$,

- Set $\mathbf{Z} = (Z(\mathbf{s}_1), \ldots, Z(\mathbf{s}_n))^T$.
- Set $\mathbf{c} = \big(C(\mathbf{s}_0 - \mathbf{s}_1), \ldots, C(\mathbf{s}_0 - \mathbf{s}_n)\big)^T$.
- Define $\mathbf{C}$ by $C_{ij} = C(\mathbf{s}_i - \mathbf{s}_j)$.
- Set $m = \dfrac{1 - \sum_i (\mathbf{C}^{-1}\mathbf{c})_i}{\sum_i \sum_j (\mathbf{C}^{-1})_{ij}}$.
- Set $\boldsymbol{\lambda} = \mathbf{C}^{-1}(\mathbf{c} + \mathbf{m})$, where $\mathbf{m} = (m, m, \ldots, m)^T_{1\times n}$.
- Set $\sigma_k^2(\mathbf{s}_0) = C(\mathbf{0}) - \boldsymbol{\lambda}^T\mathbf{c} + m$.
  - The ordinary kriging predictor of $\mathbf{Z}(\mathbf{s}_0)$ is $\widehat{p}(\mathbf{Z}; \mathbf{s}_0) = \boldsymbol{\lambda}^T\mathbf{Z}$.
  - The 95% prediction interval is

$$A = [\hat{Z}(\mathbf{s}_0) - 1.96\sigma_k(\mathbf{s}_0), \hat{Z}(\mathbf{s}_0) + 1.96\sigma_k(\mathbf{s}_0)].$$

The following is an S-plus implementation of this algorithm for an isotropic covariance with parameters θ_1 and θ_2.

Let `data` be an $n \times 3$ matrix of the coordinates and the values of the data points. That is, each row represents one (x, y, z) point. Since we use isotropic correlation functions, the covariogram depends only on the distance between the points, but not on their locations, and we can calculate the matrix $\mathbf{C}$ in this way:

```
distance <- function (data, i, j)
                                    #distance between i and j locations
    sqrt ((data[i, 1]-data[j,1])^2+(data[i, 2]-data[j,2])^2)


makeC <- function (data, cov, theta1, theta2, tau){
   n <- length (data)/3
   C <- matrix (rep(0, n^2), ncol=n)
   for (i in 1:n)
     for (j in 1:n)
        C[i,j] <- cov (distance(data, i, j), theta1, theta2, tau)
   C
}
```

Here `cov` is the covariance function, and `theta1, theta2`, and `tau` are its parameters.

Let (`s0x`, `s0y`) be the coordinates of the point $\mathbf{s}_0$ where we want to predict. Then the function `makec` constructs the vector $\mathbf{c}$:

```
makec <- function (s0x, s0y, data, cov, theta1, theta2, tau){
  n <- length(data)/3
  c <- numeric(n)
  for (i in 1:n)
    c[i] <- cov (sqrt((s0x-data[i, 1])^2 + (s0y-data[i, 2])^2),
                                            theta1, theta2, tau)
  c(c)
}
```

The function `kriging` performs the prediction at the points (x, y) with coordinates stored in the vectors `x` and `y`. Here the `filename` is a name of the file in which we want to put the results.

```
kriging <- function(x, y, data, cov, theta1, theta2, tau, filename){
        n <- length(data)/3
        Cinv <- solve(makeC(data, cov, theta1, theta2, tau))
        sumCinv <- sum(Cinv)
        lx <- length(x)
        z <- numeric(lx)
        for(i in 1:lx) {
          s0x <- x[i]
          s0y <- y[i]
          vectorc <- makec(s0x, s0y, data, cov, theta1, theta2, tau)
          m <- (1 - sum(Cinv %*% vectorc))/sumCinv
          lambda <- c((vectorc + m) %*% Cinv)
          z[i] <- sum(lambda * data[1:n, 3])
        }
        if(lx == 1) {
             sigmak <- sqrt(1/tau - sum(lambda * vectorc) + m)
             list(left = z[1] - 1.96 * sigmak, prediction = z[1],
                                        right = z[1] + 1.96 * sigmak)
        }
        else write.table(cbind(x, y, z), file = filename, sep = " ",
                        dimnames.write = F)
}
```

Usually we want to make a prediction not just at one point, but on a grid. Then we want to construct `x`, the vector of all the `x`-coordinates of the grid points, and `y`, the vector of all the `y`-coordinates. For example, if we want to have a 50×50 grid with `x`-coordinates from 200 to 249 and `y`-coordinates from 50 to 99, the function `makegrid` will give us the corresponding `x` and `y` vectors:

```
makegrid <- function () {
     x <- numeric(2500)
     for (i in 1:50)
         x[((i-1)*50 + 1) : (i*50)] <- rep (200 + i - 1, 50)
     y <- numeric(2500)
     for (i in 1:50)
         y[((i-1)*50 + 1) : (i*50)] <- 50:99
     cbind (x , y)
}
```

For example if we want to do ordinary kriging with the exponential correlation function, $\theta_1 = \exp(-0.03)$, $\theta_2 = 0.5$, and variance $1/\tau = 1$, taking data from the file "Sample" of the form

x	y	z
1	2	3.356
2	5	4.383
1	3	2.751
	$\ldots$,	

we proceed like this:

```
expcov <- function (l, theta1, theta2, tau){
          theta1^(l^theta2)/tau
}
data <- matrix(scan("Sample", byrow=T, ncol=3)
grid <- makegrid()
kriging (grid[,1], grid[,2], data, expcov,
                          exp(-0.03), 0.5, 1, "KrigResult")
```

Figures 2.1, 2.2, and 2.3 show results of applying the kriging algorithm. First we generated a Gaussian random field (Figure 2.1), than sampled 50 random points from it, and used them as data for kriging. Figure 2.2 shows the resulting kriging surface. Figure 2.3 shows the the original Gaussian field overlaid with its kriging prediction. In all the figures $\tau = 1$.

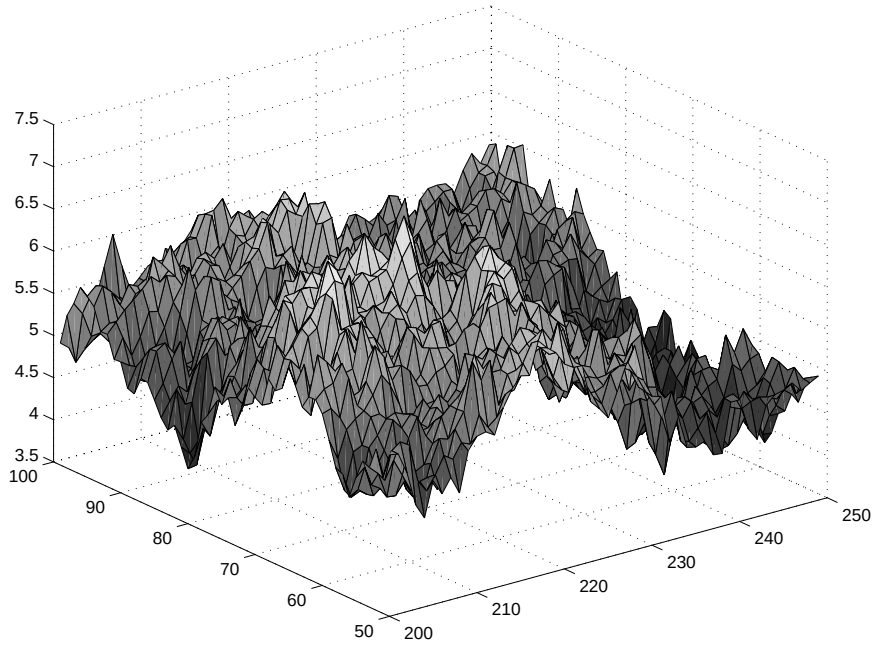

Figure 2.1: Gaussian(5, 1) random field, exponential(exp(-0.03), 1) correlation.

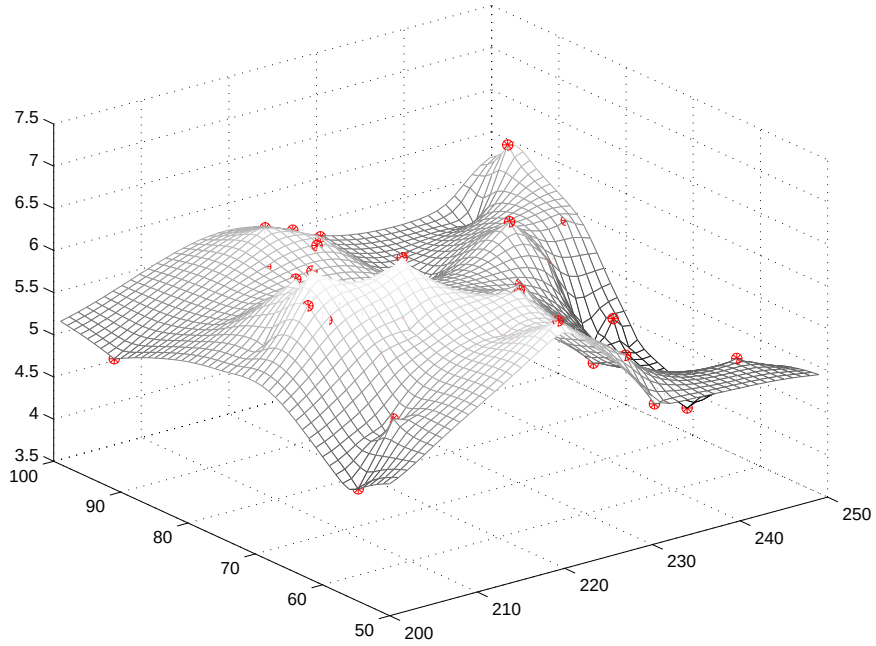

Figure 2.2: Kriging surface obtained from 50 data points.

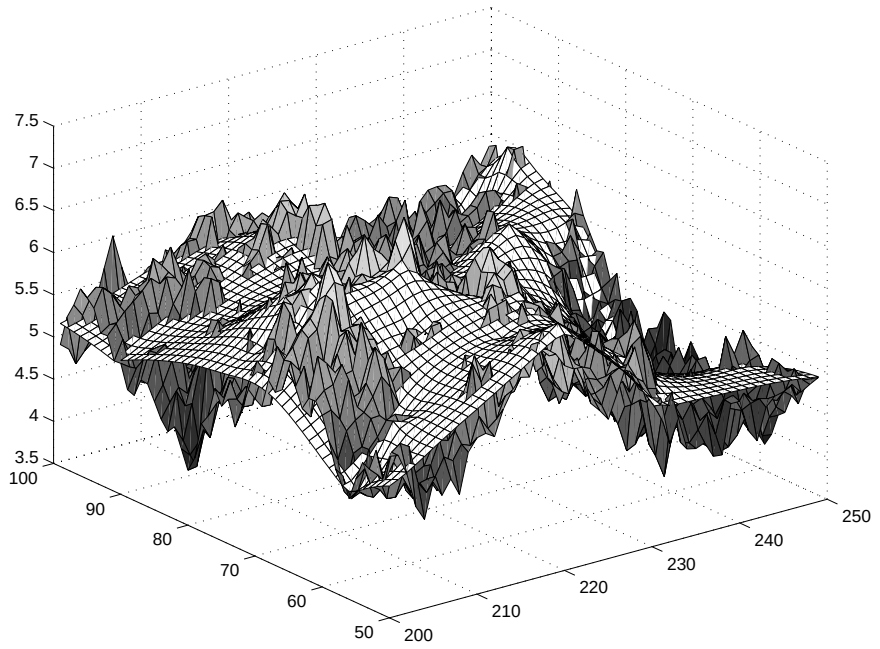

Figure 2.3: Overlay of a Gaussian(5, 1) random field with exponential (exp(-0.03), 1) correlation and its kriging approximation from 50 data points.

2.2 Trans-Gaussian Kriging

Suppose now that the $Z(.)$ process is obtained from

$$Z(\mathbf{s}) = \phi(Y(\mathbf{s})), \quad \mathbf{s} \in D, \tag{2.9}$$

where $Y(.)$ is a stationary Gaussian process and ϕ is a one-to-one twice-differentiable function. The idea is to transform the problem from the Z scale to the Y scale, predict $Y(\mathbf{s}_0)$ and transform the result back. The problem with this approach is that it gives a *biased* predictor.

Using the δ-method around $\mu_Y = E(Y(\mathbf{s}))$ we can obtain a bias correction.

The ordinary kriging predictor is unbiased and so

$$E(Y(\mathbf{s}_0)) = E(\hat{Y}(\mathbf{s}_0)) = \mu_Y.$$

Define

$$\check{Z}(\mathbf{s}_0) = \check{p}_Z(\mathbf{Z}; \mathbf{s}_0) = \phi(\hat{p}_Y(\mathbf{Y}; \mathbf{s}_0)) - B,$$

where $\hat{p}_Y(\mathbf{Y}; \mathbf{s}_0)$ is a regular kriging predictor on the $\mathbf{Y}$ scale and B is a constant, such that the predictor is unbiased:

$$E(\check{Z}(\mathbf{s}_0)) = E(Z(\mathbf{s}_0)).$$

We shall now find B such that the predictor is unbiased.

Expanding ϕ around μ_Y we get

$$Z = \phi(Y) \approx \phi(\mu_Y) + (Y - \mu_Y)\phi'(\mu_Y) + \frac{(Y - \mu_Y)^2}{2}\phi''(\mu_Y).$$

Therefore,

$$E(Z) \approx \phi(\mu_Y) + \frac{\phi''(\mu_Y)}{2} E\left((Y - \mu_Y)^2\right)$$

$$= \phi(\mu_Y) + \frac{\phi''(\mu_Y)}{2}\text{Var}(Y). \tag{2.10}$$

Similarly, if we denote $Z(\mathbf{s}_0)$ by Z_0

$$\begin{aligned}
E(\check{Z}_0) &= E(\phi(\hat{Y}_0)) - B \\
&\approx E\left(\phi(\mu_Y) + (\hat{Y}_0 - \mu_Y)\phi'(\mu_Y) + \frac{(\hat{Y}_0 - \mu_Y)^2}{2}\phi''(\mu_Y)\right) - B \\
&= \phi(\mu_Y) + (E(\hat{Y}_0) - \mu_Y)\phi'(\mu_Y) + \frac{\phi''(\mu_Y)}{2}E\left((\hat{Y}_0 - \mu_Y)^2\right) - B \\
&= \phi(\mu_Y) + \frac{\phi''(\mu_Y)}{2}E\left((\hat{Y}_0 - \mu_Y)^2\right) - B
\end{aligned} \tag{2.11}$$

From (2.10) and (2.11), the unbiasedness condition becomes

$$\phi(\mu_Y) + \frac{\phi''(\mu_Y)}{2}\text{Var}(Y) = \phi(\mu_Y) + \frac{\phi''(\mu_Y)}{2}E\left((\hat{Y}_0 - \mu_Y)^2\right) - B.$$

So the bias B is

$$B = \frac{\phi''(\mu_Y)}{2}E\left((\hat{Y}_0 - \mu_Y)^2\right) - \frac{\phi''(\mu_Y)}{2}\text{Var}(Y). \tag{2.12}$$

It remains to compute $E\left((\hat{Y}_0 - \mu_Y)^2\right)$. By definition (2.2),

$$\hat{Y}_0 = \boldsymbol{\lambda}_Y^T \mathbf{Y}.$$

Therefore,

$$\begin{aligned}
E\left((\hat{Y}_0 - \mu_Y)^2\right) &= E\left((\boldsymbol{\lambda}_Y^T\mathbf{Y} - \mu_Y)^2\right) \\
&= E\left((\boldsymbol{\lambda}_Y^T\mathbf{Y})^2 - 2\boldsymbol{\lambda}_Y^T\mathbf{Y}\mu_Y + {\mu_Y}^2\right) \\
&= E\left(\left(\sum_{i=1}^n \lambda_i Y_i\right)^2\right) - 2\mu_Y E\left(\sum_{i=1}^n \lambda_i Y_i\right) + {\mu_Y}^2 \\
&= \sum_i \sum_j \lambda_i \lambda_j E(Y_i Y_j) - {\mu_Y}^2 \\
&= \sum_i \sum_j \lambda_i \lambda_j (\mathbf{C}_{ij} + {\mu_Y}^2) - {\mu_Y}^2
\end{aligned}$$

$$= \sum_i \sum_j \lambda_i \lambda_j \mathbf{C}_{ij}, \quad \text{because} \sum_i \sum_j \lambda_i \lambda_j {\mu_Y}^2 = {\mu_Y}^2$$
$$= \boldsymbol{\lambda}_Y^T \mathbf{C}_Y \boldsymbol{\lambda}_Y.$$

From (2.12),

$$B = \phi''(\mu_Y) \left(\frac{\boldsymbol{\lambda}_Y^T \mathbf{C}_Y \boldsymbol{\lambda}_Y}{2} - \frac{\text{Var}(Y)}{2} \right).$$

Since $\mathbf{C}\boldsymbol{\lambda} = \mathbf{c} + \mathbf{m}$ (see (2.5)),

$$\begin{aligned} B &= \phi''(\mu_Y) \left(\frac{\boldsymbol{\lambda}_Y^T (\mathbf{c}_Y + \mathbf{m}_Y)}{2} - \frac{\text{Var}(Y)}{2} \right) \\ &= \phi''(\mu_Y) \left(\frac{\boldsymbol{\lambda}_Y^T \mathbf{c}_Y + \boldsymbol{\lambda}_Y^T \mathbf{m}_Y}{2} - \frac{\text{Var}(Y)}{2} \right) \\ &= \phi''(\mu_Y) \left(\frac{\boldsymbol{\lambda}_Y^T \mathbf{c}_Y + m_Y}{2} - \frac{\text{Var}(Y)}{2} \right) \\ &= \phi''(\mu_Y) \frac{\boldsymbol{\lambda}_Y^T \mathbf{c}_Y + m_Y - C(\mathbf{0})}{2} \\ &= \phi''(\mu_Y) \frac{-(C(\mathbf{0}) - \boldsymbol{\lambda}_Y^T \mathbf{c}_Y - m_Y)}{2} \end{aligned}$$

Recall that from (2.8),

$$\sigma_k^2(\mathbf{s}_0) = C(\mathbf{0}) - \boldsymbol{\lambda}^T \mathbf{c} + m.$$

Then

$$\begin{aligned} B &= \phi''(\mu_Y) \frac{-(\sigma_{k,Y}^2(\mathbf{s}_0) - m_Y - m_Y)}{2} \\ &= -\phi''(\mu_Y) \left(\frac{\sigma_{k,Y}^2(\mathbf{s}_0)}{2} - m_Y \right). \end{aligned}$$

Therefore, the approximately unbiased predictor is

$$\check{p}_{\mathbf{Z}}(\mathbf{Z}; \mathbf{s}_0) = \phi(\hat{p}_Y(\mathbf{Y}; \mathbf{s}_0)) + \phi''(\mu_Y) \left(\frac{\sigma_Y^2(\mathbf{s}_0)}{2} - m_Y \right),$$

Note, that we do not know μ_Y, therefore we use its MLE instead. The likelihood of μ is given by

$$\begin{aligned}
L(\mu|\mathbf{Y}) &= \frac{1}{\sqrt{2\pi}|\mathbf{C}|}\exp\left(-\frac{1}{2}(\mathbf{Y}-\mu)^T\mathbf{C}^{-1}(\mathbf{Y}-\mu)\right)\\
\log L(\mu|\mathbf{Y}) &= \log\left(\frac{1}{\sqrt{2\pi}|\mathbf{C}|}\right) - \frac{1}{2}(\mathbf{Y}-\mu)^T\mathbf{C}^{-1}(\mathbf{Y}-\mu)\\
&= \log\left(\frac{1}{\sqrt{2\pi}|\mathbf{C}|}\right) - \frac{1}{2}\sum_i\sum_j(Y_i-\mu)(\mathbf{C}^{-1})_{ij}(Y_j-\mu)\\
&= \log\left(\frac{1}{\sqrt{2\pi}|\mathbf{C}|}\right) - \frac{1}{2}\sum_i\sum_j(\mu^2-\mu(Y_i+Y_j)+Y_iY_j)(\mathbf{C}^{-1})_{ij}.
\end{aligned}$$

Differentiating, we get

$$\frac{\partial(\log L)}{\partial\mu} = -\frac{1}{2}\sum_i\sum_j(2\mu - Y_i - Y_j)(\mathbf{C}^{-1})_{ij}.$$

Equating to zero,

$$\begin{aligned}
2\mu\sum_i\sum_j(\mathbf{C}^{-1})_{ij} &= \sum_i\sum_j(Y_i+Y_j)(\mathbf{C}^{-1})_{ij}\\
\mu &= \frac{1}{2}\frac{\sum_i\sum_j(Y_i+Y_j)(\mathbf{C}^{-1})_{ij}}{\sum_i\sum_j(\mathbf{C}^{-1})_{ij}}\\
&= \frac{\sum_i(\mathbf{C}^{-1}\mathbf{Y})_i}{\sum_i\sum_j(\mathbf{C}^{-1})_{ij}} \qquad (2.13)
\end{aligned}$$

So, the final form of the trans-Gaussian predictor is

$$\check{p}_{\mathbf{Z}}(\mathbf{Z};\mathbf{s}_0) = \phi(\hat{p}_Y(\mathbf{Y};\mathbf{s}_0)) + \phi''(\hat{\mu}_Y)\left(\frac{\sigma_Y^2(\mathbf{s}_0)}{2} - m_Y\right),$$

where

$$\begin{aligned}
\hat{\mu}_Y &= \frac{\sum_i(\mathbf{C_Y}^{-1}\mathbf{Y})_i}{\sum_i\sum_j(\mathbf{C_Y}^{-1})_{ij}} \qquad \text{(from (2.13))},\\
m_Y &= \frac{1-\sum_i(\mathbf{C_Y}^{-1}\mathbf{c}_Y)_i}{\sum_i\sum_j(\mathbf{C_Y}^{-1})_{ij}},\\
\sigma_Y^2(\mathbf{s}_0) &= C(\mathbf{0}) - \boldsymbol{\lambda}_Y^T\mathbf{c}_Y + m_Y.
\end{aligned}$$

By definition, the mean squared prediction error of $\check{Z}(\mathbf{s}_0)$ is

$$\begin{aligned}
\sigma^2_{k,Z}(\mathbf{s}_0) &= E(\check{Z}(\mathbf{s}_0) - Z(\mathbf{s}_0))^2 \\
&= E\left(\phi(\hat{Y}(\mathbf{s}_0)) - B - \phi(Y(\mathbf{s}_0))\right)^2 \\
&\approx E\left(\phi(\mu_Y) + (\hat{Y}(\mathbf{s}_0) - \mu)\phi'(\mu_Y) - B - \phi(\mu_Y) - (Y(\mathbf{s}_0) - \mu_Y)\phi'(\mu_Y)\right)^2 \\
&= E\left(\phi'(\mu_Y)(\hat{Y}(\mathbf{s}_0) - Y(\mathbf{s}_0)) - B\right)^2 \\
&= E((\phi'(\mu_Y)(\hat{Y}(\mathbf{s}_0) - Y(\mathbf{s}_0)))^2) - 2B(\phi'(\mu_Y)E(\hat{Y}(\mathbf{s}_0) - Y(\mathbf{s}_0)) + B^2 \\
&= (\phi'(\mu_Y))^2\sigma^2_{k,Y}(\mathbf{s}_0) + B^2, \qquad \text{since } \hat{Y} \text{ is unbiased.}
\end{aligned}$$

For small values of the bias B the mean squared prediction error of $\check{Z}(\mathbf{s}_0)$ is approximately given by (Cressie, p. 137)

$$\sigma^2_{k,Z}(\mathbf{s}_0) \approx (\phi'(\hat{\mu}_Y))^2\sigma^2_{k,Y}(\mathbf{s}_0).$$

Indeed, in the calculations B^2 took values in the range of 2% - 4% of the value of $(\phi'(\hat{\mu}_Y))^2\sigma^2_{k,Y}(\mathbf{s}_0)$.

The 95% prediction interval is approximately

$$A = [\check{Z}(\mathbf{s}_0) - 1.96\sigma_{k,Z}(\mathbf{s}_0), \check{Z}(\mathbf{s}_0) + 1.96\sigma_{k,Z}(\mathbf{s}_0)].$$

Therefore, the algorithm for the trans-Gaussian kriging is as follows.

Given the values $Z(\mathbf{s}_1), \ldots, Z(\mathbf{s}_n)$ at points $\mathbf{s}_1, \ldots, \mathbf{s}_n$, the covariance function $C(.)$, the transformation ϕ such that $\mathbf{Z}(\mathbf{s}) = \phi(\mathbf{Y}(\mathbf{s}))$ for a Gaussian field $\mathbf{Y}$, and the location of interest $\mathbf{s}_0$,

- Set `dataZ`$= (Z(\mathbf{s}_1), \ldots, Z(\mathbf{s}_n))^T$.
- Construct the new dataset `dataY` as ϕ inverse of `dataZ`.
- Predict $\mathbf{Y}(\mathbf{s}_0)$ using ordinary kriging for `dataY`.

- Take ϕ of the result and add the correction term:

$$\check{p}_{\mathbf{Z}}(\mathbf{Z};\mathbf{s}_0) = \phi(\hat{p}_Y(\mathbf{Y};\mathbf{s}_0)) + \phi''(\hat{\mu}_Y)\left(\frac{\sigma_Y^2(\mathbf{s}_0)}{2} - m_Y\right).$$

To make the prediction on a grid, again, we need first to construct the grid using the function `makegrid`, read the data into the matrix `dataZ`, and use the function `transkriging`:

```
 dataZ <- matrix(scan("Sample", byrow=T, ncol=3)

transkriging <- function(x, y, data, cov, theta1, theta2, tau,
                                                lambda, filename) {
  n <- length(data)/3
  dataY <- c(phiinverse (data[1:n, 3], lambda))
  C <- makeC(data, cov, theta1, theta2, tau)
  Cinv <- solve(C)
  sumCinv <- sum (Cinv)
  muY <- sum (Cinv %*% dataY) / sumCinv
  lx <- length(x)
  z <- numeric(lx)
  for (i in 1:lx){
      s0x <- x[i]
      s0y <- y[i]
      vectorc <- makec(s0x, s0y, data, cov, theta1, theta2, tau)
      num <- sum(Cinv %*% vectorc)
      mY <- (1 - num) / sumCinv
      boldlambda <- c((vectorc + mY) %*% Cinv)
      sigma2 <- 1/tau - sum(boldlambda * vectorc) + mY
```

```
      z[i] <- phi(sum(boldlambda * dataY), lambda) +
              phidouble (muY, lambda) * (sigma2/2 - mY)
 }
 write.table(cbind(x,y,z), file = filename, sep = " ",
                                          dimnames.write = F)
 if (lx==1)  {
    sigmak <- phiprime(muY, lambda) * sqrt(sigma2)
    list (left=z[1] - 1.96*sigmak, prediction=z[1],
                                    right=z[1] + 1.96*sigmak)
 }
}
```

Actually, we use trans-Gaussian kriging for $\phi \equiv \phi_\lambda$, where ϕ_λ is from the family $\{\phi_\lambda, \lambda \in \Lambda\}$ and `lambda` in the program is the parameter of this family. Therefore `phidouble` is the second derivative of the function ϕ_λ, and `phiinv` is the inverse of ϕ_λ.

2.3 Box-Cox Transformation

The family of transformations that we discuss here is the Box-Cox family (Box and Cox, 1964): for $x > 0$

$$g_\lambda(x) = \begin{cases} \dfrac{x^\lambda - 1}{\lambda}, & \text{for } \lambda \neq 0 \\ \log x, & \text{for } \lambda = 0. \end{cases}$$

Our trans-Gaussian model (2.9) is

$$Z(\mathbf{s}) = \phi(Y(\mathbf{s})), \mathbf{s} \in D,$$

where $Y(.)$ is a stationary Gaussian process and ϕ is a twice-differentiable function. Assumes in what follows that the Box-Cox transformation of the data gives a Gaussian field. That is,

$$\phi^{-1}(Z(\mathbf{s})) = g_\lambda(Z(\mathbf{s}))$$

is a Gaussian random field. We use $\phi(x) = \phi_\lambda(x)$. Note that

$$\phi(x) = \begin{cases} (x\lambda + 1)^{\frac{1}{\lambda}} & \text{, for } \lambda \neq 0, \\ \exp(x) & \text{, for } \lambda = 0. \end{cases}$$

$$\phi(x)' = \begin{cases} (x\lambda + 1)^{\frac{1}{\lambda}-1} & \text{, for } \lambda \neq 0, \\ \exp(x) & \text{, for } \lambda = 0. \end{cases}$$

$$\phi(x)'' = \begin{cases} \left(\frac{1}{\lambda} - 1\right) \lambda (x\lambda + 1)^{\frac{1}{\lambda}-2} & \text{, for } \lambda \neq 0, \\ \exp(x) & \text{, for } \lambda = 0. \end{cases}$$

The family of distributions obtained by the inverse Box-Cox transformation of a Gaussian field includes the Gaussian distribution ($\lambda = 1$) and the lognormal distribution ($\lambda = 0$).

Note that the inverse Box-Cox transformation $\phi(.)$ can result in a tremendous increase of variance. For example, let Y be $N(5, 1)$ and Z be such that the Box-Cox transformation of Z is Gaussian, that is $Z = \phi(Y)$. Then, for $\lambda = 1$, $Z \sim N(6, 1)$ and so $E(Z) = 6$, $\text{Var}(Z) = 1$. For $\lambda = 0.5$,

$$\begin{aligned} E(Z) &= E\left((\lambda Y + 1)^{\frac{1}{\lambda}}\right) \\ &= E\left(\left(\frac{1}{2}Y + 1\right)^2\right) \\ &= \int \left(\frac{1}{2}y + 1\right)^2 f(y)dy \end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{4}\int y^2 f(y)dy + \int y f(y)dy + \int f(y)dy \\
&= \frac{1}{4}E(Y^2) + E(Y) + 1 \\
&= \frac{1}{4} \times 26 + 5 + 1 \\
&= 12.5 \\
E(Z^2) &= E\left(\left((\lambda Y + 1)^{\frac{1}{\lambda}}\right)^2\right) \\
&= E\left(\left(\frac{1}{2}Y + 1\right)^4\right) \\
&= \int \left(\frac{1}{2}y + 1\right)^4 f(y)dy \\
&= \frac{1}{16}\int y^4 f(y)dy + \frac{1}{2}\int y^3 f(y)dy + \\
&\quad \frac{3}{2}\int y^2 f(y)dy + 2\int y f(y)dy + \int f(y)dy \\
&= \frac{1}{16}E(Z^4) + \frac{1}{2}E(Z^3) + \frac{3}{2}E(Y^2) + 2E(Y) + 1 \\
&= \frac{1}{16} \times 778 + \frac{1}{2} \times 140 + \frac{3}{2} \times 26 + 2 \times 5 + 1 \\
&= 168.625 \\
\mathrm{Var}(Z) &= 168.625 - (12.5)^2 = 12.375
\end{aligned}
$$

For $\lambda = 0$, Z is lognormal and

$$E(Z) = \exp\left(\mu + \frac{1}{2}\sigma^2\right) = \exp\left(5 + \frac{1}{2}1\right) = 244.69$$

$$
\begin{aligned}
\mathrm{Var}(Z) &= \exp(2(\mu + \sigma^2)) - \exp(2\mu + \sigma^2) \\
&= \exp(12) - \exp(11) = 102,880.6
\end{aligned}
$$

Figures 2.4 - 2.6 demonstrate how the field changes under the inverse Box-Cox transformation.

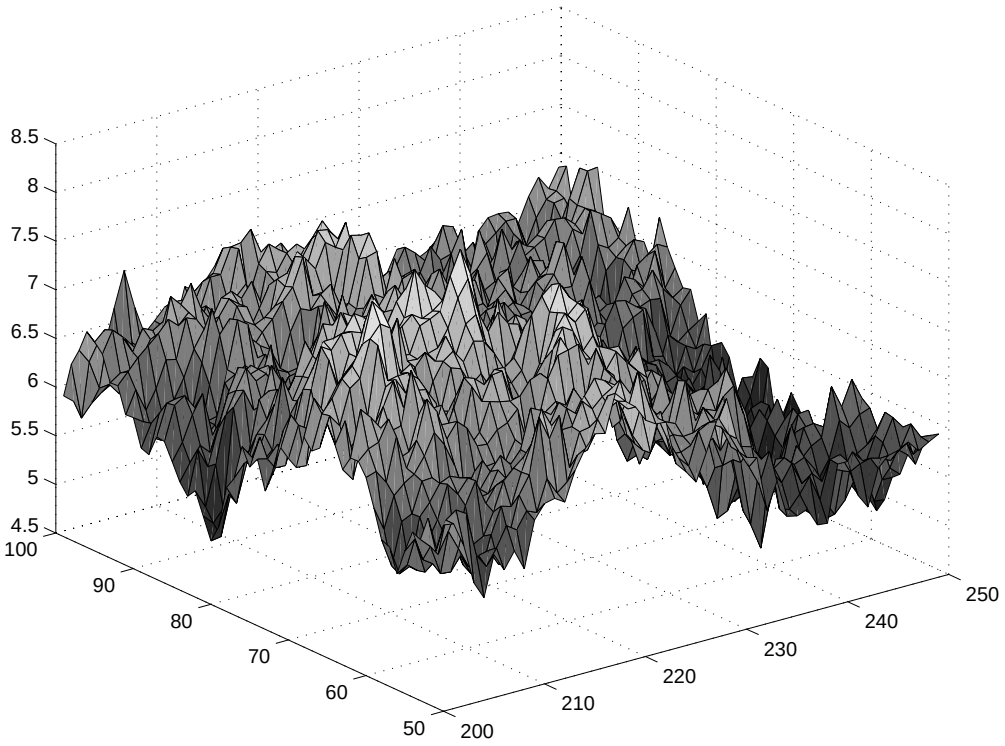

Figure 2.4: Inverse Box-Cox transformation with $\lambda = 1$ of the Gaussian(5, 1) random field with exponential (exp(-0.03), 1) correlation. This is a Gaussian(6, 1) field.

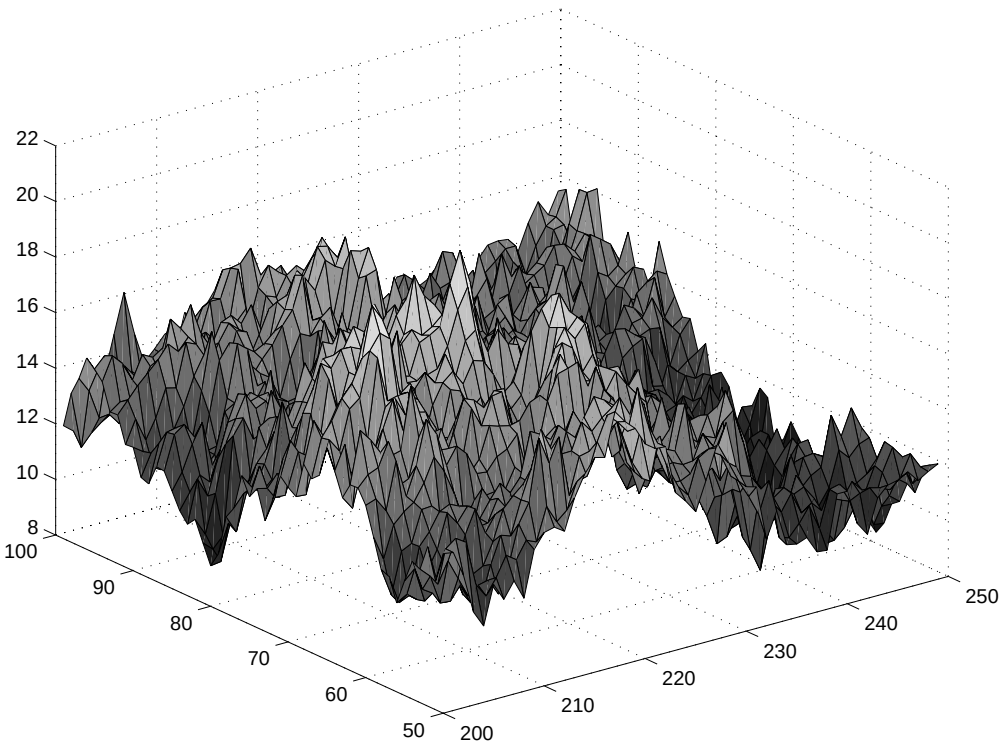

Figure 2.5: Inverse Box-Cox transformation with $\lambda = 0.5$ of the Gaussian(5, 1) random field with exponential (exp(-0.03), 1) correlation; mean = 12.5, variance = 12.375

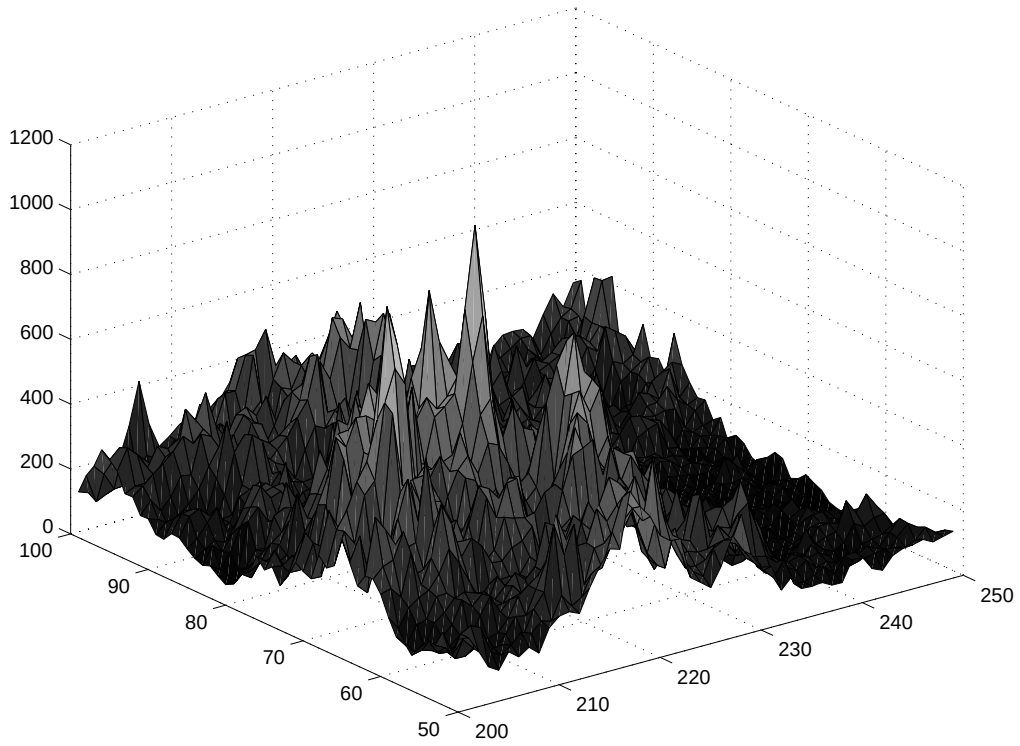

Figure 2.6: Inverse Box-Cox transformation with $\lambda = 0$ of the Gaussian(5, 1) random field with exponential (exp(-0.03), 1) correlation, that is a Lognormal field; mean = 244.69, variance = 102,880.6

Chapter 3

Bayesian Transformed Gaussian model

3.1 Introduction

The standard kriging approach to prediction in transformed Gaussian field has some potential drawbacks. First, little is said about how to identify the normalizing transformation, and in practice the logarithmic transformation is often chosen by default (*lognormal kriging*). But some other transformation may be better suited for this purpose. It is well known, at least for the Box-Cox family, that few atypical observations, or even a single one, may lead to a selection of an incorrect transformation, which would produce a bias in the resulting predictors (Atkinson and Shephard, 1996). Second, trans-Gaussian kriging and its variants may significantly underestimate the prediction uncertainty, producing overly optimistic prediction intervals.

In this chapter we describe a Bayesian method of prediction in transformed Gaussian random fields due to De Oliveira, 1997, where the transformation is known to belong to a certain parametric family of monotone transformations, but the parameter is unknown.

3.2 Model Description

Let $\{Z(\mathbf{s}), \mathbf{s} \in D\}, D \subseteq \mathbb{R}^2$ be the random field of interest, and suppose that we have n observations $\mathbf{Z} = (Z(\mathbf{s}_1), Z(\mathbf{s}_2), \ldots, Z(\mathbf{s}_n))^T$ from a single realization of this field, where $\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n$ are known distinct locations in D. Based on $\mathbf{Z}$ and our prior knowledge about the random field, we want to predict the unobserved $\mathbf{Z}_0 = \mathbf{Z}(\mathbf{s}_0)$, where $\mathbf{s}_0$ is a given location in D. It is assumed that $\mathbf{Z}_0$ comes from the same realization as the data vector $\mathbf{Z}$.

Let $\mathcal{G} = \{g_\lambda : \lambda \in \Lambda\}$ be a parametric family of transformations where each $g_\lambda \in \mathcal{G}$ is a nonlinear monotone transformation, such that $(\partial/\partial\lambda)g_\lambda$ exists and is continuous in $\Lambda \times \mathbb{R}$.

Our main *model assumption* is that for some unknown transformation parameter λ, the field

$$\{Y(\mathbf{s}) = g_\lambda(Z(\mathbf{s}) : \mathbf{s} \in D)\}$$

is a Gaussian random field with the following properties:

$$E\big(Y(\mathbf{s})\big) = \sum_{j=1}^{p} \beta_j f_j(\mathbf{s}) = \boldsymbol{\beta}^T \boldsymbol{f}(\mathbf{s}), \quad \mathbf{s} \in D,$$

where $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)^T \in \mathbb{R}^p$ are unknown regression parameters, $\boldsymbol{f}(\mathbf{s}) = (f_1(\mathbf{s}), f_2(\mathbf{s}), \ldots, f_p(\mathbf{s}))^T$ is a set of known location-dependent covariates, and

$$\mathrm{Cov}\{Y(\mathbf{s}), Y(\mathbf{t})\} = \frac{1}{\tau} r_{\boldsymbol{\theta}}(\mathbf{s} - \mathbf{t}); \quad \mathbf{s}, \mathbf{t} \in D.$$

As before, τ is the precision of the random field, $\tau^{-1} = \mathrm{Var}\big(Y(\mathbf{s})\big)$, and $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_q)^T \in \Theta \subset \mathbb{R}^q$ is a structural parameter controlling the range of correlation and/or the smoothness of the random field Y. The correlation function $r_{\boldsymbol{\theta}}$ is assumed continuous in $\boldsymbol{\theta}$.

The cases where either the random field Z or some known transformation of it is Gaussian are special cases of this model when the family contains only one member.

By the stated assumptions, we have

$$(\mathbf{g}_\lambda(\mathbf{Z}_0), \mathbf{g}_\lambda(\mathbf{Z}) \mid \boldsymbol{\beta}, \tau, \boldsymbol{\theta}, \lambda) \sim N_{k+n} \left(\begin{bmatrix} \mathbf{X}_0\boldsymbol{\beta} \\ \mathbf{X}\boldsymbol{\beta} \end{bmatrix}, \frac{1}{\tau} \begin{bmatrix} \mathbf{E}_{\boldsymbol{\theta}} & \mathbf{B}_{\boldsymbol{\theta}} \\ \mathbf{B}_{\boldsymbol{\theta}}^T & \mathbf{C}_{\boldsymbol{\theta}} \end{bmatrix} \right) \tag{3.1}$$

for some $\lambda \in \Lambda$ and $(\boldsymbol{\beta}, \tau, \boldsymbol{\theta})^T \in \mathbb{R}^p \times (0, \infty) \times \Theta$. For any vector $\mathbf{a} = (a_1, a_2, \ldots, a_n)^T$ we define $\boldsymbol{g}_\lambda(a)$ as

$$\mathbf{g}_\lambda(\mathbf{a}) \equiv (g_\lambda(a_1), g_\lambda(a_2), \ldots, g_\lambda(a_n))^T, \tag{3.2}$$

$\mathbf{X}$ and $\mathbf{X}_0$ are known $n \times p$ and $k \times p$ design matrices defined by

$$\mathbf{X}_{ij} = f_j(\mathbf{s}_i) \quad \text{and} \quad (\mathbf{X}_0)_{ij} = f_j(\mathbf{s}_{0i}), \tag{3.3}$$

and $\mathbf{E}_{\boldsymbol{\theta}}, \mathbf{B}_{\boldsymbol{\theta}}$, and $\mathbf{C}_{\boldsymbol{\theta}}$ are $k \times k, k \times n$, and $n \times n$ correlation matrices:

$$\begin{aligned} (\mathbf{E}_{\boldsymbol{\theta}})_{ij} &= r_{\boldsymbol{\theta}}(\|\mathbf{s}_{0i} - \mathbf{s}_{0j}\|), \\ (\mathbf{B}_{\boldsymbol{\theta}})_{ij} &= r_{\boldsymbol{\theta}}(\|\mathbf{s}_{0i} - \mathbf{s}_j\|), \text{ and} \\ (\mathbf{C}_{\boldsymbol{\theta}})ij &= r_{\boldsymbol{\theta}}(\|\mathbf{s}_i - \mathbf{s}_j\|). \end{aligned}$$

From (1.1), the likelihood of the model parameters, based on the original data $\mathbf{z} = (z_1, z_2, \ldots, z_n)^T$, where $z_i = Z(\mathbf{s}_i)$, is given by

$$L(\boldsymbol{\beta}, \tau, \boldsymbol{\theta}, \lambda; \mathbf{z}) = \left(\frac{\tau}{2\pi}\right)^{n/2} |\mathbf{C}_{\boldsymbol{\theta}}|^{-1/2} \exp\left\{-\frac{\tau}{2} (\mathbf{g}_\lambda(\mathbf{z}) - \mathbf{X}\boldsymbol{\beta})^T \mathbf{C}_{\boldsymbol{\theta}}^{-1} (\mathbf{g}_\lambda(\mathbf{z}) - \mathbf{X}\boldsymbol{\beta})\right\} J_\lambda$$

for $z_i \in g_\lambda^{-1}(\mathbb{R})$, and is 0 otherwise, where

$$J_\lambda = \prod_{i=1}^{n} |\mathbf{g}_\lambda^T(z_i)|$$

is the Jacobian of the transformation. As described in De Oliveira (1997) and De Oliveira, Kedem, and Short (1997), the full Bayesian model is used to predict Z_0. The full prior specification is given by

$$p(\boldsymbol{\beta}, \tau, \boldsymbol{\theta}, \lambda) \propto \frac{p(\boldsymbol{\theta})p(\lambda)}{\tau J_\lambda^{p/n}},$$

where $p(\boldsymbol{\theta})$ and $p(\lambda)$ are the prior marginals of $\boldsymbol{\theta}$ and λ. In particular, $p(\lambda)$ is uniform.

Assume now that $\mathbf{Z}_0$ is scalar. Then with the assumptions on $\mathbf{Z}$ this defines the joint distribution of the parameters and the unobserved location given the data,

$$p(\boldsymbol{\beta}, \tau, \boldsymbol{\theta}, \lambda, Z_0 \mid \mathbf{Z}).$$

By integrating out all the parameters we get the predictive density $p(Z_0 \mid \mathbf{Z})$, the posterior distribution of the value of interest Z_0. The predictor itself is the median of this distribution.

Because the parameters $\boldsymbol{\theta}$ and λ cannot be integrated out analytically, a Monte Carlo algorithm is used to approximate the predictive density numerically.

3.3 The `btg` program

The `btg` program is an implementation of the BTG prediction algorithm (Bindel, De Oliveira, Kedem, 1997). The `btg` program performs the following.

- Prediction of $Z(\mathbf{s}_0)$ and computation of its uncertainty.
- Plotting of the predictive density for any specified location.
- Computation of medians and 95% prediction intervals based on the above predictive pdf.

- Mapping of predictions and prediction uncertainties over a rectanlar region.
- Estimation of the transformation and correlation parameters.
- Cross-validation assessment.

The `tkbtg` is an X Window graphical interface to the `btg` program. With `tkbtg`, users can interactively manipulate program settings and access the functions of the `btg` engine.

The program and its full description can be downloaded from

`http://www.math.umd.edu/~bnk/btg_page.html` .

Some limitations on using the `btg` program:

1. The field has to be positive.

   The range of Z_0 should always be strictly positive, since the domains of the Box-Cox transformation functions are positive real numbers. If the data set contains some zeros or very small values, we can shift the field up by adding a positive constant, perform the prediction, and then shift everything back.

2. The parameters of the correlation function.

   For the appropriate scaling in the multidimensional optimization for correlation functions other than exponential, the `btg` program works not with the original parameter vector $\boldsymbol{\theta}$ of the correlation function, but with the vector $(\exp(-\theta_1), \exp(-\theta_2), \ldots, \exp(-\theta_n))$.

   Suppose that an interval A is the range of the correlation parameter θ. Then the optimization will be performed over the transformed range, $\exp(-A)$. Therefore, $\exp(-A)$ should not fall out of the machine precision range.

Because of this, parameters should be such that $\exp(-\theta) > \varepsilon$, and hence we should have $\theta < -\log \varepsilon$, where ε is the machine accuracy.

To achieve this, one can try to scale the x- and y-coordinates of the data.

For example, for the spherical(50) correlation,

$$\begin{aligned} r_{50}(l) &= 1 - \frac{3}{2}\left(\frac{l}{50}\right) + \frac{1}{2}\left(\frac{l}{50}\right)^3 \\ &= 1 - \frac{3}{2}\left(\frac{l/10}{50/10}\right) + \frac{1}{2}\left(\frac{l/10}{50/10}\right)^3 \\ &= r_5(l/10). \end{aligned}$$

Therefore, we can use spherical(5) correlation instead of spherical(50), if we move from l to $l/10$ by dividing the x- and the y-coordinates of the data by 10. In this case the optimization will be performed around the value of $\exp(-5)$, which is fine, rather than $\exp(-50)$, which would fail.

Simulations.

Figures 3.1 - 3.3 show the result of the BTG algorithm for a transformed Gaussian field.

First a Gaussian(5, 1) random field with Matérn(1, 10) correlation was generated. Then it was transformed by the inverse Box-Cox transformation with parameter $\lambda = 0.5$. The result is shown in Figure 3.1.

Then 50 random points where chosen and used as a data set for the BTG algorithm. The result of the mapping of the whole field is shown in Figure 3.2.

Figure 3.3 shows the real and predicted field together.

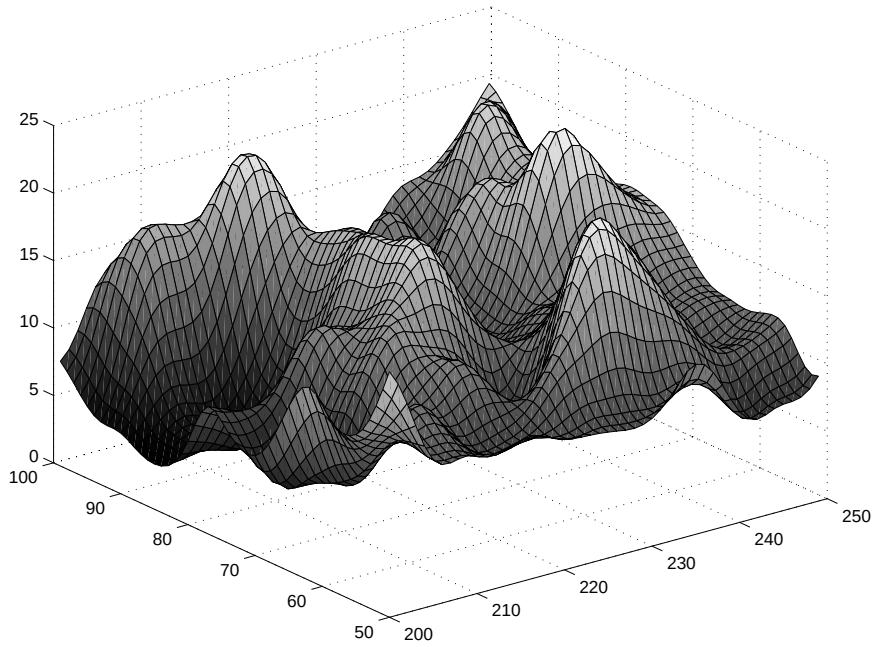

Figure 3.1: Transformed Gaussian(5,1) random field with $\lambda = 0.5$, Matérn(1,10) correlation.

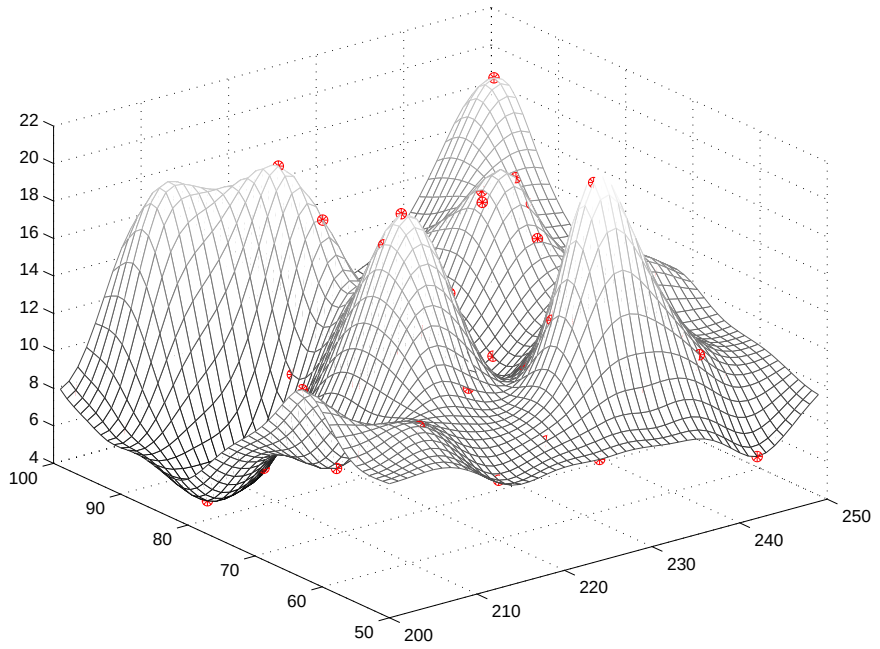

Figure 3.2: BTG surface obtained from 50 data points

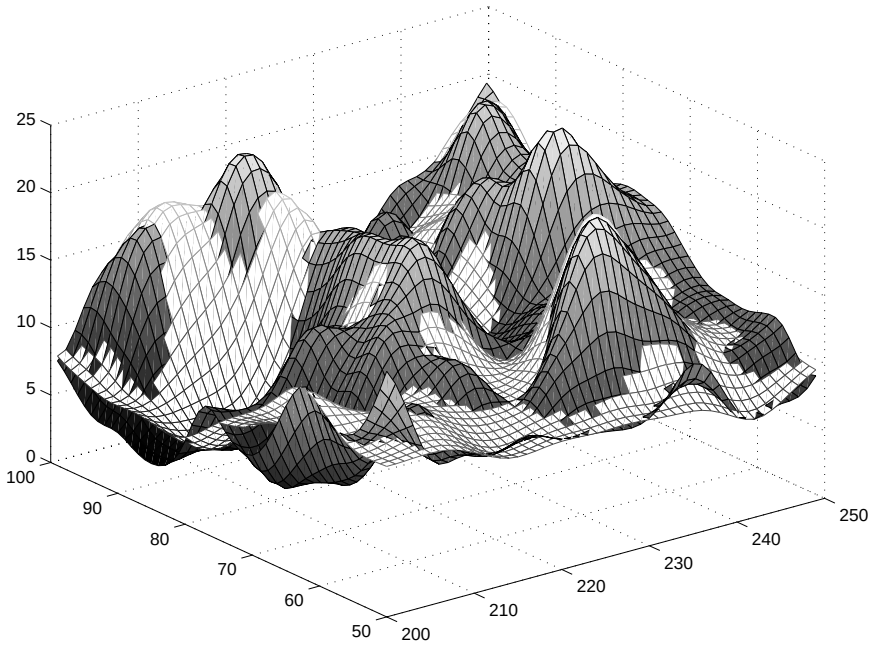

Figure 3.3: Transformed Gaussian(5,1) random field with $\lambda = 0.5$ and its BTG approximation from 50 data points.

Chapter 4

Gaussian Field Simulation

4.1 The Problem

In this chapter we describe the method that we use to generate stationary Gaussian random fields. We use such fields as simulated data in Chapter 5 in the comparison of the prediction methods.

Suppose that we want to generate a sample from a Gaussian field with mean zero and a given correlation function r over an $n_1 \times n_2$ grid $\mathbf{S} = \{\mathbf{s}_{ij}\}$. This means that we need to generate $n_1 n_2$ jointly normal random variables. We combine them in a vector $\mathbf{Z}$ row-wise, starting at the upper left corner of the grid. The grid together with the covariance function define the covariance matrix $\mathbf{C}$, so that the vector that we want to generate is $\mathbf{Z} \sim N(\mathbf{0}_{n_1 n_2}, \mathbf{C}_{n_1 n_2 \times n_1 n_2})$.

The standard way to do this is to use the Cholesky decomposition of $\mathbf{C}$,

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

where $\mathbf{L}$ is a lower triangular matrix. The cost of this decomposition is $(n_1 n_2)^3/6$ multiplications and $n_1 n_2$ square roots. Once $\mathbf{L}$ is obtained, we can set

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

where $\boldsymbol{\varepsilon}$ is a vector of independent identically distributed $N(0,1)$ random variables. Then indeed $\mathbf{Z}$ is normal and

$$E(\mathbf{Z}) = E(\mathbf{L}\boldsymbol{\varepsilon}) = \mathbf{L}E(\boldsymbol{\varepsilon}) = \mathbf{0},$$
$$\mathrm{Var}(\mathbf{Z}) = \mathbf{L}\mathrm{Var}(\boldsymbol{\varepsilon})\mathbf{L}^T = \mathbf{L}\mathbf{L}^T = \mathbf{C}.$$

This approach has the following problems:

- The storage of the $n_1 n_2 \times n_1 n_2$ matrix $\mathbf{C}$ is required.
- The Cholesky decomposition of $\mathbf{C}$ is too expensive computationally.

Because of these difficulties we use the Circulant Embedding method instead (Dietrich and Newsam, 1993).

4.2 Circulant Embedding Method

The Circulant Embedding method is applicable for generating realizations of stationary Gaussian fields over regular grids (the grid is regular if it has constant x and y steps). Under these conditions, the covariance matrix of the resulting vector $\mathbf{Z} \sim N(\mathbf{0}_{n_1 n_2}, \mathbf{C})$ is block Toeplitz with Toeplitz blocks (Zimmerman, 1989). By definition, a regular Toeplitz matrix has constant values along all the diagonals, as in

$$C = \begin{pmatrix} 1 & 5 & 6 & 7 \\ 2 & 1 & 5 & 6 \\ 3 & 2 & 1 & 5 \\ 4 & 3 & 2 & 1 \end{pmatrix}.$$

A block Toeplitz matrix with Toeplitz blocks is defined as a block matrix in which the Toeplitz structure is applicable to the whole blocks and also within each block.

Related to Toeplitz matrices is the class of circulant matrices. A circulant matrix is a particular case of the Toeplitz matrix, in which all the columns starting with the second one are obtained by circulation of the previous column (a circulation is a cyclical permutation of the vector which moves its last element to the first position and shifts all the others down by one), as in

$$V = \begin{pmatrix} 1 & 4 & 3 & 2 \\ 2 & 1 & 4 & 3 \\ 3 & 2 & 1 & 4 \\ 4 & 3 & 2 & 1 \end{pmatrix}.$$

Similarly to the Toeplitz case, a block circulant matrix with circulant blocks is defined as a block matrix in which the circulant structure applies to the whole blocks and also within each block.

The idea of the Circulant Embedding method is to embed a block Toeplitz matrix into a larger block circulant matrix with circulant blocks, using the standard embedding procedure, that we illustrate in the following example. Suppose we have the symmetric block Toeplitz matrix

$$\mathbf{C} = \left(\begin{array}{ccc|ccc} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 1 & 2 & 7 & 4 & 5 \\ 3 & 2 & 1 & 8 & 7 & 4 \\ \hline 4 & 7 & 8 & 1 & 2 & 3 \\ 5 & 4 & 7 & 2 & 1 & 2 \\ 6 & 5 & 4 & 3 & 2 & 1 \end{array}\right) = \begin{pmatrix} \mathbf{C}^{(1)} & \mathbf{C}^{(2)} \\ \mathbf{C}^{(2)^T} & \mathbf{C}^{(1)} \end{pmatrix}$$

First, we embed each of the two blocks $\mathbf{C}^{(1)}$ and $\mathbf{C}^{(2)}$ into $\mathbf{V}^{(1)}$ and $\mathbf{V}^{(2)}$ respectively. The first column of $\mathbf{V}^{(1)}$ is obtained as follows: write out all the entries of the first column of $\mathbf{C}^{(1)}$, a zero, and all the entries of the first row of $\mathbf{C}^{(1)}$ backwards (without repeating C_{11}). The rest of the $\mathbf{V}^{(1)}$ is then defined by circulation:

$$\mathbf{V}^{(1)} = \text{embed}\left(\mathbf{C}^{(1)}\right) = \begin{pmatrix} 1 & 2 & 3 & 0 & 3 & 2 \\ 2 & 1 & 2 & 3 & 0 & 3 \\ 3 & 2 & 1 & 2 & 3 & 0 \\ 0 & 3 & 2 & 1 & 2 & 3 \\ 3 & 0 & 3 & 2 & 1 & 2 \\ 2 & 3 & 0 & 3 & 2 & 1 \end{pmatrix}$$

The matrix $\mathbf{V}^{(2)}$ is obtained from $\mathbf{C}^{(2)}$ in the same way.

$$\mathbf{V}^{(2)} = \text{embed}\left(\mathbf{C}^{(2)}\right) = \begin{pmatrix} 4 & 5 & 0 & 6 & 8 & 7 \\ 7 & 4 & 5 & 0 & 6 & 8 \\ 08 & 7 & 4 & 5 & 0 & 6 \\ 0 & 8 & 7 & 4 & 5 & 0 \\ 6 & 0 & 8 & 7 & 4 & 5 \\ 5 & 6 & 0 & 8 & 7 & 4 \end{pmatrix}$$

Finally, we combine the blocks $\mathbf{V}^{(i)}$ into the matrix $\mathbf{V}$ like that (note that everything is actually defined by the first block column):

$$\mathbf{V} = \begin{pmatrix} \mathbf{V}^{(1)^T} & \mathbf{V}^{(2)} & \mathbf{0}_{6\times 6} & \mathbf{V}^{(2)^T} \\ \mathbf{V}^{(2)^T} & \mathbf{V}^{(1)^T} & \mathbf{V}^{(2)} & \mathbf{0}_{6\times 6} \\ \mathbf{0}_{6\times 6} & \mathbf{V}^{(2)^T} & \mathbf{V}^{(1)^T} & \mathbf{V}^{(2)} \\ \mathbf{V}^{(2)} & \mathbf{0}_{6\times 6} & \mathbf{V}^{(2)^T} & \mathbf{V}^{(1)^T} \end{pmatrix}$$

One can see that building $\mathbf{V}$ from the blocks is analogous to embedding the individual blocks. Namely, one creates the first block column by putting together the following:

- transposed embedded blocks from the first block column of the original matrix $\mathbf{C}$;
- a block matrix of zeros;
- the embedded blocks from the first block row of the original matrix $\mathbf{C}$, in the reversed order.

The resulting $\mathbf{V}$ is

$$\mathbf{V} = \left(\begin{array}{cccccc|cccccc|cccccc|cccccc}
1&2&3&0&3&2 & 4&5&0&6&8&7 & 0&0&0&0&0&0 & 4&7&8&0&6&5 \\
2&1&2&3&0&3 & 7&4&5&0&6&8 & 0&0&0&0&0&0 & 5&4&7&8&0&6 \\
3&2&1&2&3&0 & 8&7&4&5&0&6 & 0&0&0&0&0&0 & 6&5&4&7&8&0 \\
0&3&2&1&2&3 & 0&8&7&4&5&0 & 0&0&0&0&0&0 & 0&6&5&4&7&8 \\
3&0&3&2&1&2 & 6&0&8&7&4&5 & 0&0&0&0&0&0 & 8&0&6&5&4&7 \\
2&3&0&3&2&1 & 5&6&0&8&7&4 & 0&0&0&0&0&0 & 7&8&0&6&5&4 \\
\hline
4&7&8&0&6&5 & 1&2&3&0&3&2 & 4&5&6&0&8&7 & 0&0&0&0&0&0 \\
5&4&7&8&0&6 & 2&1&2&3&0&3 & 7&4&5&6&0&8 & 0&0&0&0&0&0 \\
6&5&4&7&8&0 & 3&2&1&2&3&0 & 8&7&4&5&6&0 & 0&0&0&0&0&0 \\
0&6&5&4&7&8 & 0&3&2&1&2&3 & 0&8&7&4&5&6 & 0&0&0&0&0&0 \\
8&0&6&5&4&7 & 3&0&3&2&1&2 & 6&0&8&7&4&5 & 0&0&0&0&0&0 \\
7&8&0&6&5&4 & 2&3&0&3&2&1 & 5&6&0&8&7&4 & 0&0&0&0&0&0 \\
\hline
0&0&0&0&0&0 & 4&7&8&0&6&5 & 1&2&3&0&3&2 & 4&5&6&0&8&7 \\
0&0&0&0&0&0 & 5&4&7&8&0&6 & 2&1&2&3&0&3 & 7&4&5&6&0&8 \\
0&0&0&0&0&0 & 6&5&4&7&8&0 & 3&2&1&2&3&0 & 8&7&4&5&6&0 \\
0&0&0&0&0&0 & 0&6&5&4&7&8 & 0&3&2&1&2&3 & 0&8&7&4&5&6 \\
0&0&0&0&0&0 & 8&0&6&5&4&7 & 3&0&3&2&1&2 & 6&0&8&7&4&5 \\
0&0&0&0&0&0 & 7&8&0&6&5&4 & 2&3&0&3&2&1 & 5&6&0&8&7&4 \\
\hline
4&5&6&0&8&7 & 0&0&0&0&0&0 & 4&7&8&0&6&5 & 1&2&3&0&3&2 \\
7&4&5&6&0&8 & 0&0&0&0&0&0 & 5&4&7&8&0&6 & 2&1&2&3&0&3 \\
8&7&4&5&6&0 & 0&0&0&0&0&0 & 6&5&4&7&8&0 & 3&2&1&2&3&0 \\
0&8&7&4&5&6 & 0&0&0&0&0&0 & 0&6&5&4&7&8 & 0&3&2&1&2&3 \\
6&0&8&7&4&5 & 0&0&0&0&0&0 & 8&0&6&5&4&7 & 3&0&3&2&1&2 \\
5&6&0&8&7&4 & 0&0&0&0&0&0 & 7&8&0&6&5&4 & 2&3&0&3&2&1
\end{array}\right)$$

The main advantage of circulant matrices is the following Diagonalization Theorem (Nott and Wilson, 1997): if $\mathbf{V}$ is block circulant with circulant blocks, then

$$\mathbf{V} = \mathbf{F}^H \mathbf{\Lambda} \mathbf{F}, \tag{4.1}$$

where $\mathbf{\Lambda}$ is the diagonal matrix of the eigenvalues of $\mathbf{V}$, and $\mathbf{F}$ is the matrix of the two-dimensional Fourier transform defined by

$$F_{lm} = \frac{1}{\sqrt{n_1 n_2}} \exp\left(-\frac{2\pi i}{n_1}\left[\frac{l}{n_2}\right]\left[\frac{m}{n_2}\right]\right) \exp\left(-\frac{2\pi i}{n_2}(l \bmod n_2)(m \bmod n_2)\right)$$
$$l, m = 0, 1, \ldots, n_1 n_2 - 1$$

The reason $\mathbf{F}$ is called the two-dimensional Fourier transform matrix is that for any vector $\mathbf{a}$ the product $\mathbf{Fa}$ can be computed as

$$\mathbf{Fa} = \text{vector}(\text{FFT}_2(\text{matrix}(\mathbf{a}))), \tag{4.2}$$

where matrix($\mathbf{a}$) denotes the $n_1 \times n_2$ matrix obtained from the vector $\mathbf{a}$ row-wise, and vector($\mathbf{A}$) is the inverse of this operation, that is, the $n_1 n_2$-vector obtained row-wise from the matrix $\mathbf{A}$. Finally, FFT_2 is the two-dimensional Fourier transform, which operates on rectangular matrices, and for an $n_1 \times n_2$ matrix $\mathbf{A}$ returns the matrix $\mathbf{B}$ of the same size with the entries

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

The main importance of (4.2) is that the cost of FFT_2 is $n_1 n_2 \log_2(n_1 n_2)$, at least in the ideal case of both n_1 and n_2 being powers of two. On the other hand, the cost of computing $\mathbf{Fa}$ for an arbitrary matrix $\mathbf{F}$ is $(n_1 n_2)^2$.

From (4.1) it follows that $\boldsymbol{\lambda}$, the vector of the eigenvalues of $\mathbf{V}$, is given by $\sqrt{n_1 n_2}\,\mathbf{F}\mathbf{V}_1$, where $\mathbf{V}_1$ is the first column of $\mathbf{V}$:

$$\mathbf{F}\mathbf{V} = \boldsymbol{\Lambda}\mathbf{F} \quad (\text{from (4.1), because } \mathbf{F} \text{ is unitary})$$

$$\mathbf{F}\mathbf{V}_1 = \boldsymbol{\Lambda}\mathbf{F}_1 \quad (\text{consider the first columns})$$

$$\mathbf{F}\mathbf{V}_1 = \frac{1}{\sqrt{n_1 n_2}}\boldsymbol{\lambda} \quad \left(\text{since } \mathbf{F}_1 = (1/\sqrt{n_1 n_2}, 1/\sqrt{n_1 n_2}, \ldots, 1/\sqrt{n_1 n_2})^T\right).$$

In the Circulant Embedding method, this observation is an analog of the Cholesky decomposition.

Now for a non-negative definite symmetric block circulant matrix $\mathbf{V}$ with circulant blocks we can generate two independent realizations $\mathbf{W}^{(1)}, \mathbf{W}^{(2)} \sim N(\mathbf{0}, \mathbf{V})$ by setting

$$\delta_i, \varepsilon_i \sim N(0, \lambda_i),\ i = 1, \ldots, n_1 n_2, \text{ mutually independent,}$$

$$\mathbf{W}^{(1)} = \mathrm{Re}(\mathbf{F}(\boldsymbol{\varepsilon} + i\boldsymbol{\delta})),$$

$$\mathbf{W}^{(2)} = \mathrm{Im}(\mathbf{F}(\boldsymbol{\varepsilon} + i\boldsymbol{\delta})).$$

Now it remains to explain how the desired sample $\mathbf{Z} \sim N(\mathbf{0}, \mathbf{C})$ is obtained. We start by embedding the symmetric non-negative definite block Toeplitz $n_1 n_2 \times n_1 n_2$ matrix $\mathbf{C}$ with Toeplitz blocks into a larger $4n_1 n_2 \times 4n_1 n_2$ block circulant matrix $\mathbf{V}$ with circulant blocks. Hopefully, the non-negative definiteness is preserved (the conditions are discussed in Kozintsev, 1999). Then we can generate $\mathbf{W}^{(1)}_{4n_1 n_2}$ and $\mathbf{W}^{(2)}_{4n_1 n_2}$ as described above. Finally, we use the components of $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$ that correspond to the original field to create the vectors $\mathbf{Z}^{(1)}$ and $\mathbf{Z}^{(2)}$ using the following procedure:

- take the first $2n_1 n_2$ elements of $\mathbf{W}$;
- divide it into $2n_1$ groups of length n_2 each;

- use every other group starting with the first one to populate an $n_1 \times n_2$ matrix row-wise; this matrix is the desired sample.

The advantages of this procedure over the Cholesky method are that only the first column of $\mathbf{C}$ has to be stored ($4n_1n_2$ values rather than $(n_1n_2)^2$), and that the cost is of the order of only $4n_1n_2 \log_2(4n_1n_2)$.

Therefore the Circulant Embedding algorithm is as follows. Given an $n_1 \times n_2$ regular grid and a covariance function,

- construct $n_1n_2 \times n_1n_2$ block Toeplitz covariance matrix $\mathbf{C}$;
- embed $\mathbf{C}$ into $4n_1n_2 \times 4n_1n_2$ block circulant matrix $\mathbf{V}$;
- generate $\mathbf{W}^{(1)}, \mathbf{W}^{(2)} \sim N(\mathbf{0}_{4n_1n_2}, \mathbf{V})$;
- extract $\mathbf{Z}^{(1)}, \mathbf{Z}^{(2)} \sim N(\mathbf{0}_{n_1n_2}, \mathbf{C})$ from $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$.

The program (Kozintsev, 1999) implementing the Circulant Embedding algorithm is available at `http://www.math.umd.edu/~bnk/bak/generate.cgi`.

Figures 4.1 - 4.4 demonstrate the results of generating Gaussian random fields with different correlation functions.

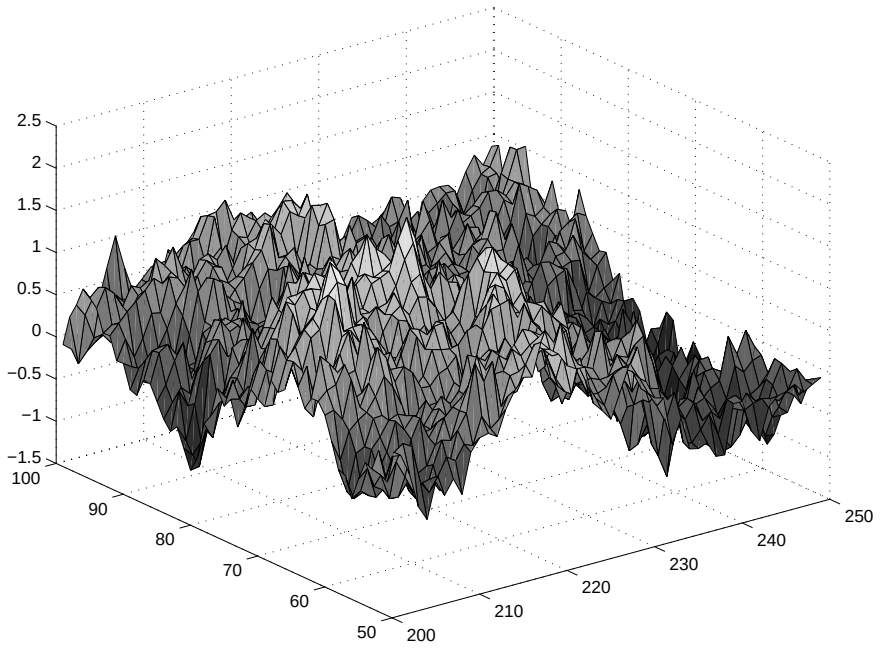

Figure 4.1: Gaussian(0, 1) r.f. with exponential (exp(-0.03), 1) correlation.

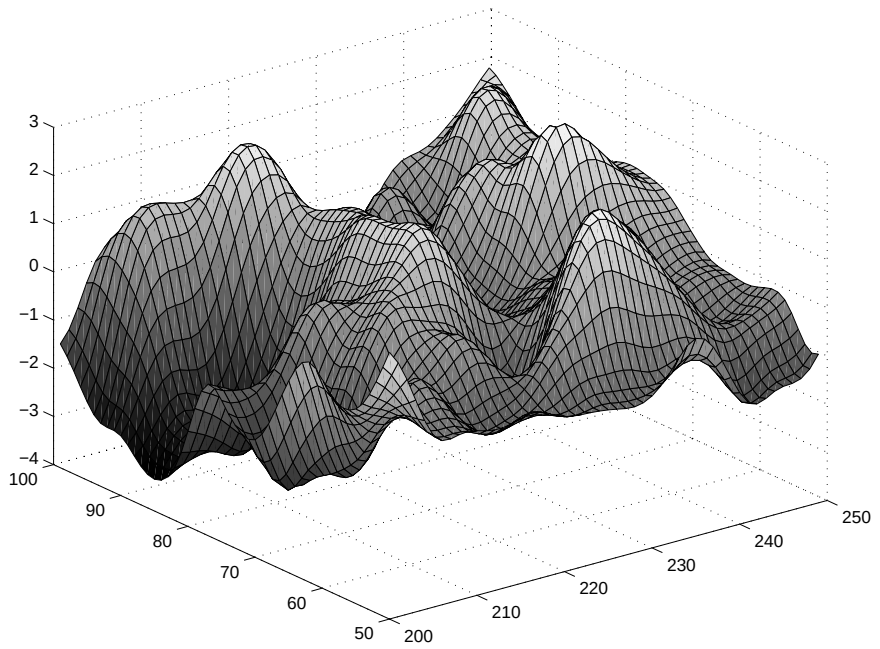

Figure 4.2: Gaussian(0, 1) r.f. with Matérn(1, 10) correlation.

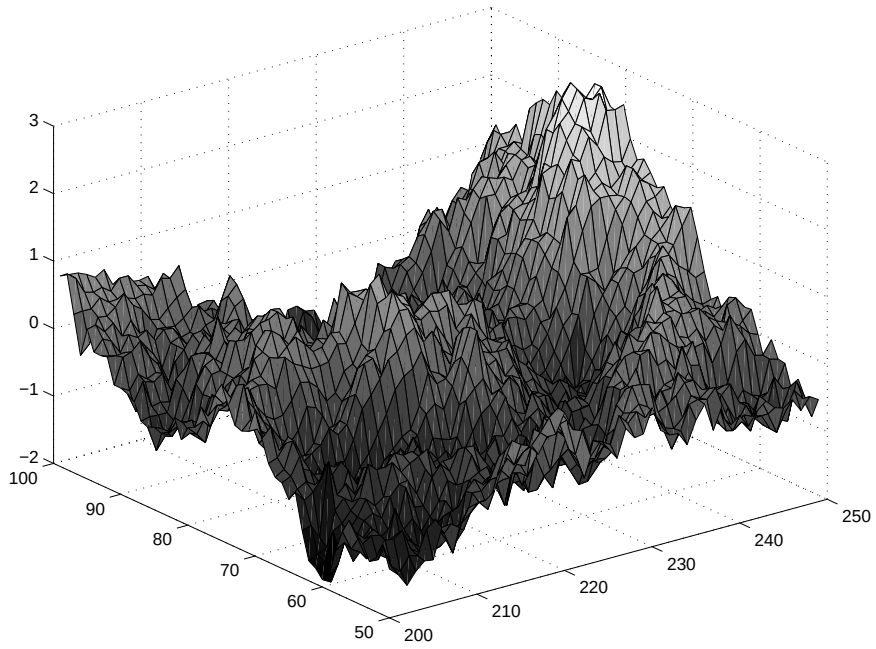

Figure 4.3: Gaussian(0, 1) r.f. with spherical(50) correlation.

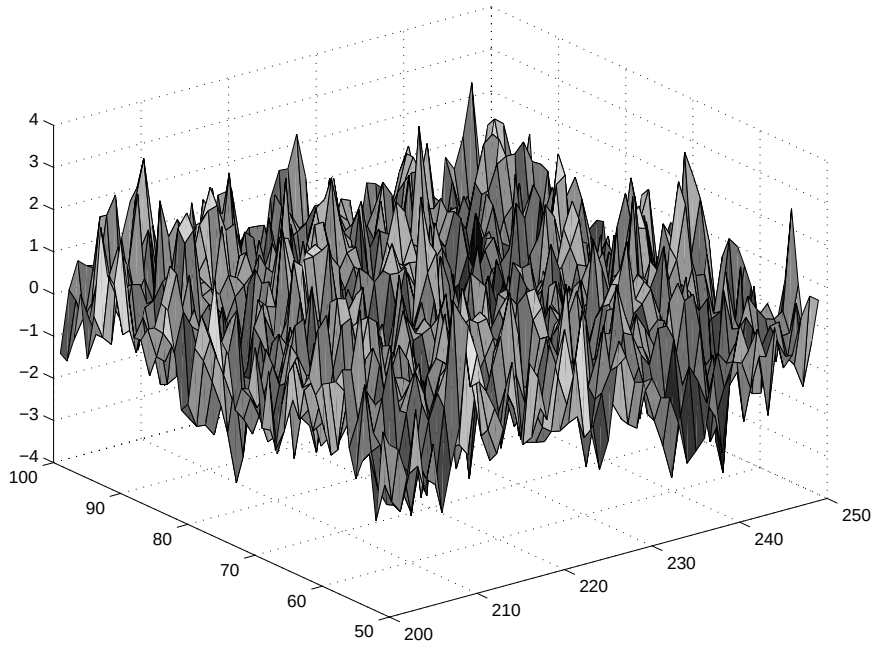

Figure 4.4: Gaussian(0, 1) r.f. with rational quadratic(0.9, 1) correlation.

Chapter 5

Comparison

In this chapter we compare the performances of the three prediction methods: ordinary kriging, trans-Gaussian kriging, and BTG. We consider artificially generated data and real rainfall (instantaneous radar reflectivity) data separately.

5.1 Generated Data

We use the following procedure for the comparison.

1. Generate a Gaussian random field Y on a 50 by 50 grid. Throughout this chapter we use Gaussian fields with mean 5, variance 1, and will every time specify the correlation function.

2. Sample 50 random points from Y.

3. Take an inverse Box-Cox transformation of Y with fixed parameter λ. This gives us the field of interest Z, and in particular the inverse Box-Cox transformation of the 50 sampled points are the given data $\mathbf{Z}$.

4. Perform the prediction of all 2500 points of the grid given the 50 data points. Compare the mean squared error (this is possible only for the

generated data).

5. Perform cross validation. That is, throw away one data point at a time and predict it using the remaining 49 points. Compare the mean squared errors (this is possible for both the generated and real data). While doing the cross validation we also calculate the 95% prediction intervals and compare their average lengths and the proportion of points that lie outside their prediction intervals.

Tables 5.1 and 5.2 show the results of predicting all 2500 points.

Correlation function	Exponential $(e^{-0.03}, 1)$			Matérn (1, 10)		
Transformation parameter	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 1$
KRIG	8117.8751	1.5318	0.11331	42406.756	3.8547	0.31564
TGK	8254.7361	1.5418	0.11331	33604.421	3.8193	0.31564
BTG	7755.0195	1.5652	0.11649	39879.496	3.7506	0.31088

Table 5.1: Mean squared error for the Exponential and Matérn correlations.

The big error for $\lambda = 0$ (lognormal data) in all three methods is due to the big variance in the field (see Section 2.3 on the properties of the inverse Box-Cox transformation). These results show that there is practically no significant difference between the methods if we consider only the MSE. Sometimes BTG gave the best result (Matérn correlation, $\lambda = 0.5$), sometimes the worst (exponential correlation, $\lambda = 1$), but the ratios of the BTG results and other two methods never exceeded 1.2. One has to note, though, that ordinary kriging and trans-Gaussian kriging had much more information about the data than BTG did -

Correlation function	Rational Quadratic (0.9, 1)			Spherical (50)		
Transformation parameter	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 1$
KRIG	109303.978	12.8835	1.0516	48050.015	3.6724	0.26738
TGK	108684.558	12.8775	1.0516	48677.479	3.6429	0.26738
BTG	117419.446	12.9808	1.0525	48378.299	3.3690	0.27476

Table 5.2: Mean squared error for the Rational Quadratic and Spherical correlations.

in both kriging methods the exact values of the transformation parameter λ and the correlation parameters where provided.

Now we perform cross validation. Tables 5.3 and 5.4 show the mean squared errors, the average lengths of the 95% prediction intervals (denoted by l), and the percentages of the points that are outside of their prediction interval (denoted by 'out').

Correlation function	Exponential $(e^{-0.03}, 1)$			Matérn (1, 10)		
Transform. parameter	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 1$
KRIG	12212.3228 'out'=98% l=1.45525	1.83605 'out'=64% l=1.43314	0.13184 'out'=2% l=1.45315	68397.4887 'out'=100% l=2.4195	7.14896 'out'=48% l=2.5128	0.58113 'out'=6% l=2.4183
TGK	11974.7347 'out'=20% l=267.92	1.84176 'out'=4% l=5.24335	0.13184 'out'=2% l=1.45315	55260.9006 'out'=18% l=291.797	7.0762 'out'=8% l=8.21326	0.58113 'out'=6% l=2.4183
BTG	12520.7 'out'=6% l=466.69	1.89171 'out'=2% l=6.0971	0.13550 'out'=2% l= 1.6289	64134.3000 'out'=12% l=330.6752	7.30742 'out'=6% l=10.2343	0.55691 'out'=6% l=2.8660

Table 5.3: Cross Validation on 50 data points: exponential and Matérn correlations. The entries are MSE, average length of the 95 % PI, and the percentage of the observations outside their PI.

Correlation function	Spherical (50)			Rational Quadratic (0.9, 1)		
Transform. parameter	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 1$
KRIG	3024.651 'out'=100% l= 1.4197	1.740893 'out'=56% l= 1.4197	0.157050 'out'=6% l=1.4197	28423.78 'out'=98% l=3.9102	10.35269 'out'=56% l=3.9102	0.91438 'out'=2% l=3.9102
TGK	2886.036 'out'=12% l=151.64	1.733068 'out'=6% l=4.7351	0.157050 'out'=6% l=1.4197	28362.65 'out'=4% l=552.24	10.34515 'out'=0% l=13.590	0.91458 'out'=2% l=3.91019
BTG	2788.953 'out'=10% l= 150.369	1.68321 'out'=4% l=5.5751	0.150843 'out'=4% l=1.6409	31120.6 'out'=20% l=305.31	10.42212 'out'=2% l=14.1282	0.92135 'out'=4% l=4.34191

Table 5.4: Cross Validation on 50 data points: spherical and rational quadratic correlations. The entries are MSE, average length of the 95 % PI, and the percentage of the observations outside their PI.

From these tables we can see that there is a big difference in the accuracy of the prediction intervals. Ordinary kriging gives good results only for $\lambda = 1$, i.e. for the Gaussian data. For all the correlation functions, except for the rational quadratic, we see that in terms of giving the wrong intervals the BTG gave the same result as the trans-Gaussian kriging on the Gaussian data ($\lambda = 1$), was a little better for the data with $\lambda = 0.5$, and much better for the log-normal data ($\lambda = 0$). This is so since the BTG prediction intervals tend to be wider. Due to the difference in the variance of the fields all three methods have better accuracy of the prediction intervals for the fields closer to Gaussian. The ordinary kriging PI's are based on the Gaussian assumption, because of this PI's for $\lambda = 0$ and $\lambda = 0.5$ are unrealistically narrow and miss most of the true values.

For some reason no pattern was found on the data with rational quadratic correlation.

Up to this point we used the exact values for all the parameters in ordinary and TG kriging. In practice we do not have them. To study the effects of mis-specifying the parameters we now perform ordinary and trans-Gaussian kriging with some of the parameters fixed at values close to the true values, but not exactly equal.

Tables 5.5 and 5.6 show the results of the trans-Gaussian kriging performed with different values of the transformation parameter λ, deviating from the true λ^*.

Both times the original field has the exponential$(e^{-0.03}, 1)$ correlation. For Table 5.5, $\lambda^* = 1$ is the true λ-value and $\lambda = 1.914$ is the result of the BTG prediction. For Table 5.6, $\lambda^* = 0$ is the true λ-value and $\lambda = 0.174642$ is the result of the BTG prediction.

$\lambda = 1.914$	$\lambda = 1.5$	$\lambda = 1.2$	$\lambda^* = 1$	$\lambda = 0.8$	$\lambda = 0.5$	$\lambda = 0$
0.13104	0.13137	0.13166	0.13184	0.13194	0.13254	0.25774
'out'=64%	'out'=48%	'out'=16%	'out'=2%	'out'=0%	'out'=0%	'out'=0%
l=0.3195	l=0.6353	l=1.0441	l=1.4531	l=2.0215	l=3.3140	l=7.5355

Table 5.5: TGK with different values of λ. $\lambda^* = 1$ is the true parameter value.

$\lambda = 1$	$\lambda = 0.8$	$\lambda = 0.5$	$\lambda = 0.174642$	$\lambda^* = 0$	$\lambda = -0.1$
12212.323	12160.493	12131.410	12122.638	11974.733	12533.73
'out'=98%	'out'=98%	'out'=92%	'out'=56%	'out'=20%	'out'=4%
l=1.4531	l= 4.2326	l= 20.537	l=110.24	l=267.92	l=443.58

Table 5.6: TGK with different values of λ. $\lambda^* = 0$ is the true parameter value.

The pattern here is that for smaller values of λ the length of the prediction intervals increase and so the number of 'wrong' intervals decreases. So when λ is overestimated we have too small prediction intervals, and when λ is underestimated, the MSE becomes bigger and the length of prediction intervals increases very fast. In other words, the length of the PI's increases with λ.

Next we keep the true value of λ fixed but change the correlation parameter. Table 5.7 shows the results for the spherical(50) correlation for various fixed values of λ. Cross validation by the trans-Gaussian kriging was performed with the true value of λ and different values of the correlation parameter.

For all three choices of λ, the minimum of the MSE was achieved for the true parameter value $\theta^* = 50$. For smaller values of θ the prediction intervals became longer and the number of 'wrong' intervals smaller. The length of PI's decreases as θ increases.

		$\theta = 5$	$\theta = 25$	$\theta = 40$	$\theta^* = 50$	$\theta = 60$	$\theta = 100$	$\theta = 150$
$\lambda = 0$	MSE	118282.7	10899.2	9076.9	8276.3	8401.43	8791.52	8942.31
	'out'	8%	4%	16%	24%	32%	58%	72%
	l	848.4	388.9	254.1	192.9	l144.4	78.5	56.391
$\lambda = 0.5$	MSE	10.9178	1.43924	1.3009	1.2949	1.3130	1.3431	1.3531
	'out'	2%	0%	0%	4%	6%	14%	28%
	l	13.913	7.171	5.512	4.815	4.271	3.132	2.505
$\lambda = 1$	MSE	0.823386	0.11387	0.1019	0.1015	0.10269	0.10482	0.1055
	'out'	4%	0%	0%	4%	4%	12%	18%
	l	3.742	1.9682	1.5531	1.389	1.2683	0.9815	0.801

Table 5.7: TGK with various values of the correlation parameter. $\theta^* = 50$ is the true parameter value.

5.2 Rainfall Data

The data set analyzed in this section is formed by rainfall intensity, measured in terms of the radar reflectivity (dBZ) at 08:28 March 09, 1998 in Melbourne, FL by a ground based radar during a period when the Tropical Rain Measuring Mission (TRMM) satellite was overhead. The data consist of a 151x151 floating point array of the intensity values. Figure 5.1 shows the data overlaid on a geographical map.

The data contain -99's as missing data, all of which should be ignored, being either below minimum detectability or out of bound.

We selected a 50 by 50 area which does not have negative values. Figure 5.2 shows the shape of this area.

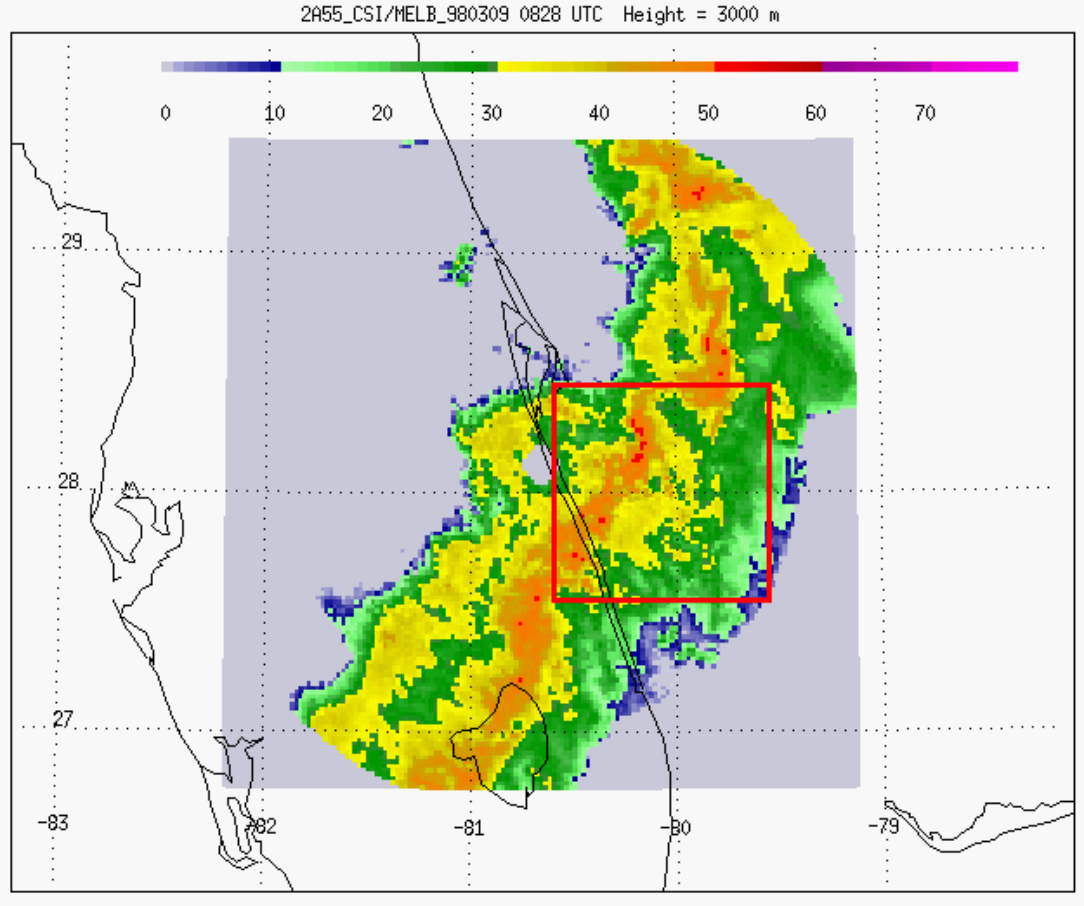

Figure 5.1: Radar reflectivity data, measured in Melbourne, FL at 08:28 March 09, 1998 with the 50×50 region used in the comparison of the prediction methods.

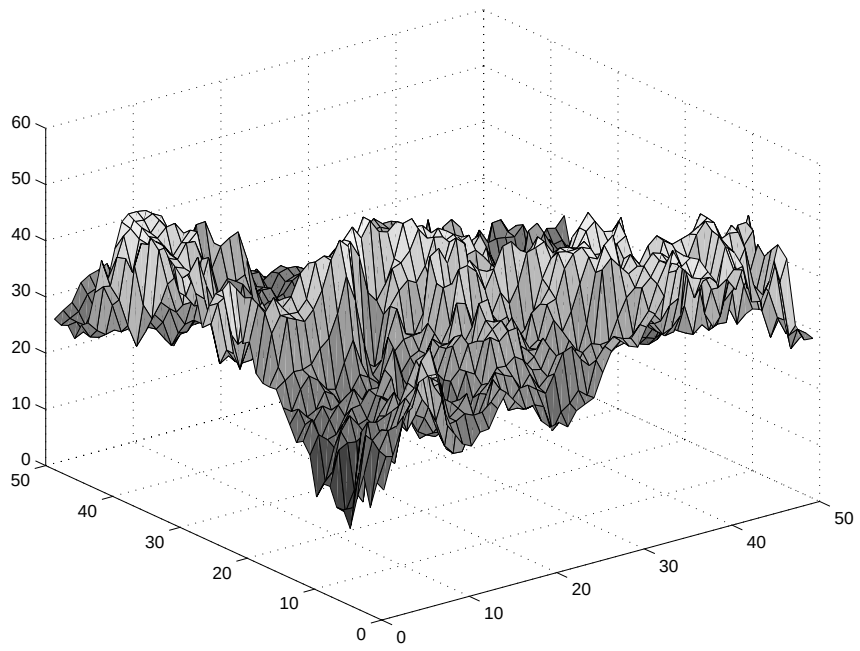

Figure 5.2: Rainfall field: Radar reflectivity in dBZ.

Then we selected two uniform random samples of 50 points each. Tables 5.8 and 5.9 show the results of the BTG cross validation on these samples. MSE in these tables is the mean squared error and 'length' is the average length of the PI's calculated over the 50 data points.

	Exponential	Rational Quadratic	Matérn	Spherical
MSE	23.9162	26.3607	32.5693	66.0787
'out'	4%	4%	8%	4%
length	22.147397	22.442944	22.602144	37.301770
lambda	0.873181	0.947889	0.914281	1.05367
theta1	0.994849	0.043728	9.20291e-08	9.83739e-09
theta2	1.04202	0.999999	0.507735	-

Table 5.8: Rainfall data. Cross validation BTG results - sample 1.

	Exponential	Rational Quadratic	Matérn	Spherical
MSE	13.5818	13.6904	14.5174	39.8341
'out'	4%	2%	4%	4%
length	18.061345	17.798766	18.800780	27.859814
lambda	0.0604596	-0.125429	0.0235469	0.180278
theta1	0.972899	0.0656118	6.86406e-07	1.85106e-08
theta2	1.04789	0.999998	0.51503	-

Table 5.9: Rainfall data. Cross validation BTG results - sample 2.

From now on only the second sample will be used, and since both samples gave the best result with the exponential correlation, we will restrict the kriging calculations only to this correlation.

5.2.1 Parameter Estimation for the Ordinary and Lognormal Kriging.

To do any type of kriging we need to know the correlation parameters θ_1 and θ_2, and precision τ. We will discuss several ways to obtain these parameters.

For a known λ Cressie (1993, p. 97) suggests the following method of obtaining θ_1, θ_2, and τ.

We know that semivariogram for the exponential correlation function is

$$\gamma_{\boldsymbol{\theta},\tau}(l) = \frac{1}{\tau}\left(1 - \theta_1^{l^{\theta_2}}\right), \quad \text{where } \boldsymbol{\theta} = (\theta_1, \theta_2)^T.$$

We can use an estimator for this semivariogram

$$2\overline{\gamma}(l) = \frac{\left(\frac{1}{|N(l)|}\sum_{N(l)} |Z(\mathbf{s}_i) - Z(\mathbf{s}_j)|^{1/2}\right)^4}{0.457 + \frac{0.494}{|N(l)|}},$$

where $N(l) = \{(\mathbf{s}_i, \mathbf{s}_j) : l - 1/2 < \|\mathbf{s}_i - \mathbf{s}_j\| < l + 1/2\}$ and $|N(l)|$ is the number of distinct sample pairs lagged by the distance l. The reasoning behind this estimator is that, for Gaussian data, $(Z(\mathbf{s}_i) - Z(\mathbf{s}_j))^2$ is a chi-squared random variable with one degree of freedom. The power transformation that makes this most Gaussian-like is the fourth root (Cressie and Hawkins, 1980), namely $|Z(\mathbf{s}_i) - Z(\mathbf{s}_j)|^{1/2}$, the square root of the absolute difference. Thus, various location estimators can be applied to $\{|Z(\mathbf{s}_i) - Z(\mathbf{s}_j)|^{1/2} : (\mathbf{s}_i, \mathbf{s}_j) \in N(l)\}$, which, when normalized for bias, yield robust (to contamination by outliers; see Hawkins and Cressie, 1984) variogram estimators.

Now weighted least squares estimates of the parameters are obtained by minimizing

$$\sum_{l \in L} |N(l)| \left(\frac{\overline{\gamma}(l)}{\gamma_{\boldsymbol{\theta},\tau}(l)} - 1\right)^2 \tag{5.1}$$

over $(\tau, \theta_1, \theta_2) \in (0, \infty) \times (0, 1) \times (0, 2]$. Joirnel and Huijbregts (1978, p.194) recommend that the fit should be only up to half the maximum possible lag (in our case $(50\sqrt{2})/2 \approx 36$) and then only using lags for which $|N(l)| > 30$. Figure 5.3 demonstrates the distribution of $|N(l)|$.

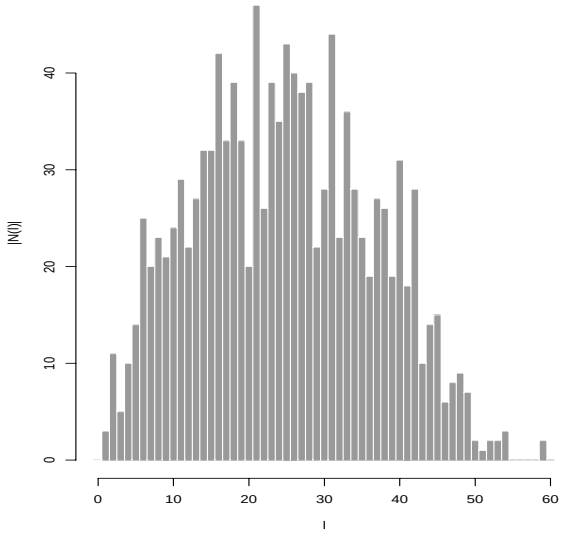

Figure 5.3: Distribution of $|N(l)|$

Ordinary kriging. Ordinary kriging means that no transformation is needed. For some reason the restriction $|N(l)| > 30$ did not work out well in this particular case. We obtained parameters from a number of different sets L and performed ordinary kriging with these parameters. The results are shown in Table 5.10.

$L = \{l : \lvert N(l)\rvert > a\}$	$a = 30$	$a = 20$	$a = 10$	$a = 0$
MSE	447.4841	14.63096	14.63174	15.06493
'out'	84%	4%	4%	2%
length	7.445913	18.444318	18.447704	19.277882
θ_1	0.99999903	0.91722721	0.92131783	0.92899560
θ_2	5.25347811	0.72268379	0.71743431	0.614514385
τ	0.01930967	0.01114444	0.01055865	0.0077932

Table 5.10: Parameter estimation for ordinary kriging: we first calculated the parameters θ_1, θ_2, and τ and then performed the cross validation using these parameters. The MSE, 'out' and 'length' are averages from the cross validation.

In this table $|N(l)| > 0$ means that we considered sum over all l less than 36. The best result was obtained for the summation over $L = \{l : |N(l)| > 20\}$.

Lognormal kriging. Now we assume that $\lambda = 0$. Table 5.11 shows the results of lognormal kriging with different parameters.

Here, again, the condition $|N(l)| > 30$ was too strong to get the best result, which was obtained for $|N(l)| > 10$.

$L = \{l : \|N(l)\| > a\}$	$a = 30$	$a = 20$	$a = 10$	$a = 0$
MSE	19.48468	15.95985	15.7621	15.83358
'out'	2%	4%	4%	4%
length	23.153523	19.008782	18.728513	18.061345
θ_1	0.9936533	0.9969230	0.9984204	0.999898834
θ_2	0.2328222	0.4415105	0.4618606	0.453506187
τ	0.2415281	0.2218373	0.1198744	0.007488926

Table 5.11: Parameter estimation for lognormal kriging: we first calculated the parameters θ_1, θ_2, and τ and then performed the cross validation using these parameters. The MSE, 'out' and 'length' are averages from the cross validation.

5.2.2 Parameter Estimation for the Trans-Gaussian Kriging.

Another common method for obtaining parameters is the maximum likelihood estimation.

We take the Box-Cox transformation of the data $\mathbf{Z}$ with unknown parameter λ. Since the result $\mathbf{Y}$ is assumed to be Gaussian$(\mu, 1/\tau)$ with exponential(θ_1, θ_2)

correlation, the likelihood function is

$$L(\lambda, \theta_1, \theta_2, \mu, \tau|\mathbf{Y}) = \frac{1}{(2\pi)^{n/2}\sqrt{|\mathbf{C}_{\boldsymbol{\theta},\tau}|}} \exp\left(-\frac{1}{2}(\mathbf{Y}-\mu\mathbf{1})^T\mathbf{C}_{\boldsymbol{\theta},\tau}^{-1}(\mathbf{Y}-\mu\mathbf{1})\right),$$

where $\boldsymbol{\theta} = (\theta_1, \theta_2)$.

Therefore the log-likelihood is

$$\begin{aligned}
&\log(L(\lambda, \theta_1,\theta_2, \mu, \tau \mid \mathbf{Y})) \\
&\qquad = \log\left(\frac{1}{(2\pi)^{n/2}\sqrt{|\mathbf{C}_{\boldsymbol{\theta},\tau}|}}\right) + \left(-\frac{1}{2}(\mathbf{Y}-\mu\mathbf{1})^T\mathbf{C}_{\boldsymbol{\theta},\tau}^{-1}(\mathbf{Y}-\mu\mathbf{1})\right) \\
&\qquad = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log(|\mathbf{C}_{\boldsymbol{\theta},\tau}|) - \left(\frac{1}{2}(\mathbf{Y}-\mu\mathbf{1})^T\mathbf{C}_{\boldsymbol{\theta},\tau}^{-1}(\mathbf{Y}-\mu\mathbf{1})\right) \\
&\qquad = -\frac{1}{2}\big(n\log(2\pi) + \log(|\mathbf{C}_{\boldsymbol{\theta},\tau}|) + (\mathbf{Y}-\mu\mathbf{1})^T\mathbf{C}_{\boldsymbol{\theta},\tau}^{-1}(\mathbf{Y}-\mu\mathbf{1})\big).
\end{aligned}$$

Unfortunately, for this particular example maximization does not work: the function has many local maxima with approximately the same value.

Another way to do the MLE is to use profiling as in Box and Cox(1964). For any fixed values of θ_1, θ_2, and λ, the ML estimators of μ_Y and τ_Y are

$$\hat{\mu}_Y = \frac{\mathbf{1}^T{\mathbf{C}_{\boldsymbol{\theta},1}}^{-1}\mathbf{g}_\lambda(\mathbf{Z})}{\mathbf{1}^T{\mathbf{C}_{\boldsymbol{\theta},1}}^{-1}\mathbf{1}},$$

where

$$\mathbf{g}_\lambda(\mathbf{a}) \equiv (g_\lambda(a_1), g_\lambda(a_2), \ldots, g_\lambda(a_n))^T,$$

and

$$\frac{1}{\hat{\tau}_Y} = \hat{\sigma}_Y^2 = \frac{1}{n}q_{\theta,\lambda}, \tag{5.2}$$

where

$$q_{\boldsymbol{\theta},\lambda} = (\mathbf{g}_\lambda(\mathbf{Z}) - \hat{\mu}_Y\mathbf{1})^T{\mathbf{C}_{\boldsymbol{\theta},1}}^{-1}(\mathbf{g}_\lambda(\mathbf{Z}) - \hat{\mu}_Y\mathbf{1}).$$

Plugging this into the log-likelihood function in the place of μ_Y and τ_Y, we obtain the profile log-likelihood, given, up to an additive constant, by

$$l(\theta_1, \theta_2, \lambda|\mathbf{Z}) = (\lambda - 1)\sum_{i=1}^{n} \log(Z_i) - \frac{n}{2}\log(q_{\theta_1,\theta_2,\lambda}) - \frac{1}{2}\log(|\mathbf{C}_{\boldsymbol{\theta},1}|). \qquad (5.3)$$

Maximizing this function with respect to θ_1, θ_2, and λ gives $\theta_1 = 0.85333, \theta_2 = 1.03889$, and $\lambda = 0.3$. Now we can plug these numbers back into the estimation of τ (5.2) and find that $\hat{\tau} = 3.3792$. The result of TGK with these parameters is shown in Table 5.12.

	$\lambda = 0.3$
MSE	15.61601
'out'	4%
length	17.14286
θ_1	0.85333
θ_2	1.03889
τ	3.379156

Table 5.12: TGK with MLE parameters. The MSE, 'out', and 'length' are averages from the cross validation.

Table 5.13 shows all the results together.

We can see that BTG produced the smallest mean squared error, while the accuracy of the prediction intervals was the same for all the methods, and the length was the smallest for the TGK, but TGK produced the largest MSE.

	ordinary kriging $\lvert N(l)\rvert > 20$	lognormal kriging $\lvert N(l)\rvert > 10$	TGK with MLE parameters	BTG
MSE	14.63096	15.7621	15.61601	13.5818
'out'	4%	4%	4%	4%
length	18.444318	18.728513	17.142864	18.061345
θ_1	0.91722721	0.9984204	0.85333	0.994849
θ_2	0.72268379	0.4618606	1.03889	1.04202
τ	0.01114444	0.1198744	3.379156	-
λ	-	0	0.3	0.0604596

Table 5.13: Summary for the exponential correlation. The MSE, 'out', and 'length' are averages obtained from the cross validation. For the first two columns we used parameters from the semivariagram approximation, and for the TGK we used the profiled MLE.

We applied the same methods to another rainfall data set - rainfall intensity was measured in Houston, TX on January 22, 1998 at 04:10. Figure 5.4 shows this data overlaid on a geographical map.

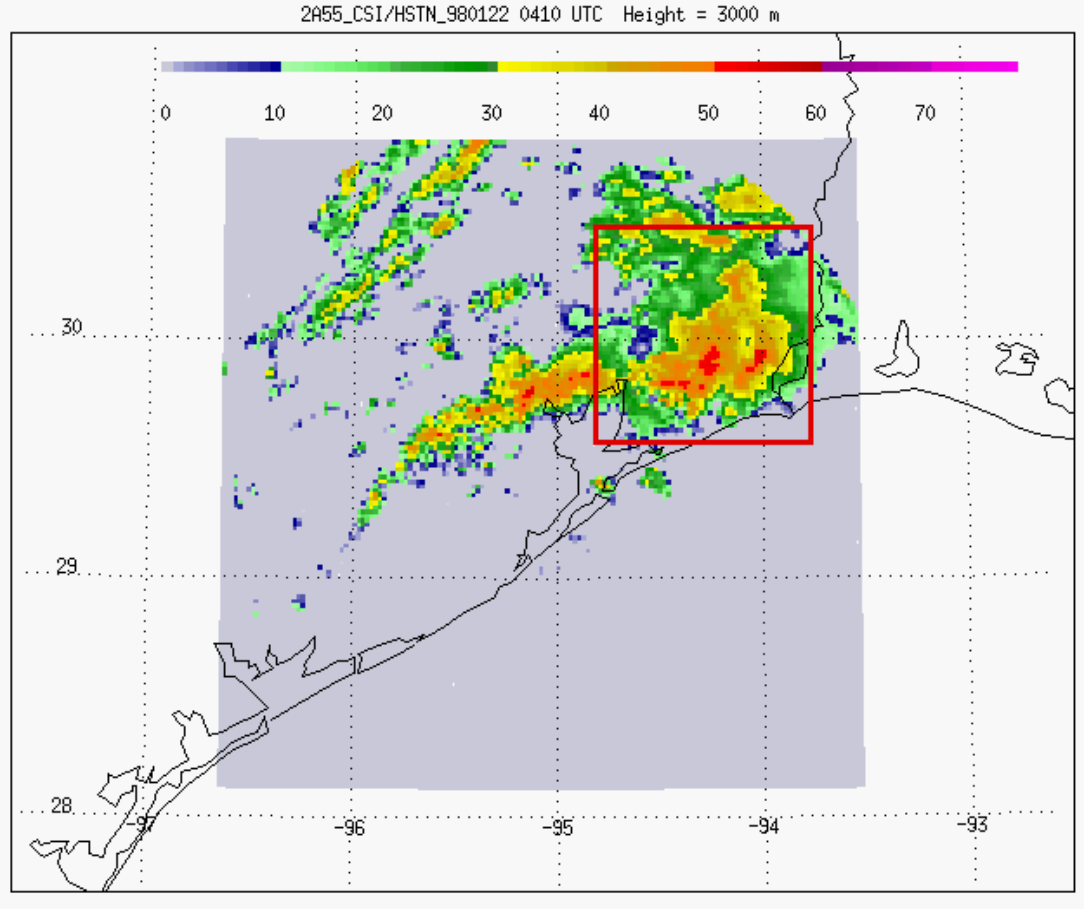

Figure 5.4: Radar reflectivity data, measured in Houston, TX at 04:10 January 22, 1998 with the 50×50 region used in the comparison of the prediction methods.

Table 5.14 shows the results of BTG with different correlation functions. Again, the best result was obtained for the exponential correlation, and so all kriging calculations where performed with this correlation. Table 5.15 summarizes all results for the Houston data set.

For this data set the ordinary kriging, TGK, and BTG gave close results. The

	Exponential	Rational Quadratic	Matérn	Spherical
MSE	59.4902	63.2864	79.2416	152.611
'out'	6%	6%	6 %	2%
length	29.961526	30.952886.	31.283445	52.641230
lambda	1.46399	1.40802	1.44646	1.34011
theta1	0.970555	0.0596284	1.957e-07	2.6136e-08
theta2	1.27101	0.999999	0.512065	-

Table 5.14: Rainfall data. Houston, TX. Cross validation BTG results.

poor result of the lognormal kriging is probably due to the fact that the data are very far from being lognormal. Indeed, both BTG and the MLE for the trans-Gaussian kriging approximated λ as being close to 1.4, while in the lognormal kriging $\lambda = 0$.

	ordinary kriging $\lvert N(l)\rvert > 0$	lognormal kriging $\lvert N(l)\rvert > 0$	TGK with MLE parameters	BTG
MSE	59.63783	119789.8	60.07092	59.4902
'out'	4%	84 %	6%	6%
length	25.614839	388.525195	30.866919	29.961526
θ_1	0.975495660	0.975495655	0.965498	0.970555
θ_2	1.395362802	1.32203625	0.821309	1.27101
τ	0.003707063	0.004421925	0.0008471929	-
λ	-	0	1.3013	1.46399

Table 5.15: Summary for the Houston data. The MSE, 'out', and 'length' are averages obtained from the cross validation. For the first two columns we used parameters from the semivariagram approximation, and for the TGK we used the profiled MLE.

5.3 Conclusions

The Gaussian (or transformed Gaussian) assumption for the spatial data involves the covariance parameters $(\boldsymbol{\theta}, \tau)$ and the transformation parameter λ.

To apply ordinary, lognormal, or trans-Gaussian kriging, one has to provide the estimates of the parameters. These estimates are not easy to obtain. Among the possible methods are the semivariogram approximation (see Section 5.2.1), and the maximization of the profiled MLE (Section 5.2.2). In both cases, one encounters difficulties with the multi-dimensional optimization in (5.1) and (5.3).

On the other hand, the BTG method does not require the specification of the parameters, since their estimation is integrated into the method. Yet, it does not fall behind the kriging techniques in terms of the mean squared error, and outperforms them in terms of the accuracy of prediction intervals, even if the kriging methods are provided with the exact true values of all the parameters. Tables 5.3 and 5.4 show that for all the four correlations the difference in the MSE never exceeded 15%, and in most cases was less than 10%.

Also for the simulated data we see that prediction intervals obtained by the BTG are more realistic. For all the cases, except the rational quadratic correlation, BTG performed as well as kriging for the Gaussian data and much better in the lognormal case – the number of points outside of their prediction intervals for BTG was at least 20% less, and for the exponential correlation three times less than that for the TGK, with only a modest increase in the interval length.

One can also see (Tables 5.5, 5.6, and 5.7) that the kriging methods are not very sensitive to parameter misspecification in terms of the mean squared error, but in terms of the prediction intervals they are. TKG is especially sensitive to the changes in the transformation parameter λ - even a slight increase in λ makes

the prediction intervals twice as short and their accuracy half as much.

Therefore, the BTG is preferable in situations when there is a large number of unknown parameters. The downside of the BTG algorithm, which employs the Monte-Carlo integration, is its computational complexity. In the setup that we used, it typically took the BTG program twice as long to predict.

From the practical point of view, we think it is useful to use both methods and compare the results.

It should be noted that this is a pilot study. For further studies, it would be helpful to obtain the confidence intervals for the parameters by using the multiple snapshots. It also might be interesting to fit more then one correlation function.

BIBLIOGRAPHY

[1] Atkinson, A. C., and Shephard, N. (1996), "Deletion Diagnostics for Transformations of Time Series," *Journal of Forecasting*, 15, 1-17.

[2] Bindel, D., De Oliveira, V., Kedem, B. (1997),"An Implementation of the Bayesian Transformed Gaussian Spatial Prediction Model",

`http://www.math.umd.edu/~bnk/btg_page.html.`

[3] Box, G. E. P., and Cox, D. R. (1964), "An Analysis of Transformations" (with discassion), *Journal of the Royal Statistical Society*, B, 26, 211-252.

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

[5] Cressie, N., and Hawkins, D. M. (1980), "Robust Estimation of the Variogram, I," *Journal of the International Association for Mathematical Geology*, 12, 115-125.

[6] Dietrich, C., and Newsam, G. (1993), "Fast and Exact Simulation of Stationary Gaussian Processes through Circulant Embedding of the Covariance Matrix," *SIAM Journal of Scientific Computations*, 18, 4, 1088-1107.

[7] Hawkins, D. M., and Cressie, N. (1984), "Robust Kriging – a Proposal," *Journal of the International Association for Mathematical Geology*, 16, 3-18.

[8] Journel, A. G., and Huijbregts, C. J. (1978), *Mining Geostatistics*, London: Academic Press,

[9] Kozintsev, B. (1999), *Computations with Gaussian Random Fields*, Ph.D. Dissertation, University of Maryland, `http://www.isr.umd.edu/TechReports/ISR/1999/PhD_99-3/PhD_99-3.phtml`.

[10] Nott, D. J., and Wilson, R. J. (1997), "Parameter Estimation for Excursion Set Texture Models", *Signal Processing*, 63, 199-210.

[11] De Oliveira, V. (1997), *Prediction in Some Classes of Non-Gaussian Random Fields*, Ph.D. Dissertation, University of Maryland, `http://www.isr.umd.edu/TechReports/ISR/1997/PhD_97-7/PhD_97-7.phtml`.

[12] 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.

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

[14] Schervish, M. J. (1995), *Theory of Statistics*, New York: Springer-Verlag.

[15] Zimmerman, D.L. (1989),"Computationally Exploitable Structure of Covariance Matrices and Generalized Covariance Matrices in Spatial Models," *Journal of Statistical Computations and Simulations*, 32, 1-15.