CARNEGIE MELLON UNIVERSITY

PREDICTING A SPACE-TIME PROCESS FROM AGGREGATE DATA EXEMPLIFIED BY THE ANIMATION OF MUMPS DISEASE

A DISSERTATION
SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
for the degree
DOCTOR OF PHILOSOPHY
in
STATISTICS

By
Audris Mockus

Department of Statistics
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

November 2004

Preface

This work, as everyting else in our universe, is an aparrent result of “ripples” or microscopic non-uniformities in the matter immediately following the Big Bang. To follow the spatial-chrolological chain of events and causal effects more closely lets consider few time-space moments (the choice and description are subjective).

1. The Big Bang at time 0, location $(0, 0, 0)$ (approximately 14 to 20 $\times10^9$ years ago). A quantum fluctation produced an object with the mass of the whole universe in a single point.

2. Time: 11.7 billion years ago (approximately 2.3 to 8.3 $\times10^9$ years). Location: Milky Way Galaxy. Event: Milky Way Galaxy Formed.

3. Time: 4.6 billion years ago (approximately 9.4 to 15.4 $\times10^9$ years). Location: Solar System. Event: Solar System Formed.

4. Time: 600 million years ago (approximately 14 to 20 $\times10^9$ years). Location: Earth. Event: abundance of life forms that left mark as first fossils.

5. Time: around 1.7 million years ago (approximately 14 to 20 $\times10^9$ years). Location: Earth. Event: Pleistocene Epoch began. This minute part of geologic time is sometimes called The Age of Man.

6. Time: 10,000 years ago. Location: North and South America. Event: First indirect record of enviromental disasters caused by humans; simultaneous advent of the Neolithic hunting tribes and demise of many Pleistocene mammals.

7. Time: Now. (approximately 14 to 20 $\times10^9$ years), location: Here. Event: nothing too important happening.

Given that the the most distant visible galaxies are more than 9×10^{22} miles away, the last four events on the list happened almost at the same location in space and almost simultaneously in time.

The vast size and great age of the universe suggest that events taking place on earth during human history are very specific to the location and the time moment. The desire of science to be general is thue crippled by this narrow window of observations in space-time. Human nature tried to counter those problems by putting itself into the center of the universe via ego-, geo-, and heliocentric theories.

Realizing the insignificance of planet Earth and everything associated with it (including humanity) I join the common effort to expand the knowledge with a tiny contribution and with the big hope of it being useful somwhere else in the great vastness of space-time.

Acknowledgements

I would like to thank my advisor William F. Eddy for his time, valuable ideas, and numerous suggestions, members of my committee Mark J. Schervish, Robert Kass, and Mark Berger for their suggestions on how to improve this document, and all the members of the statistics department for the friendly atmosphere that made this work more enjoyable for me.

I would like to thank Arthur Cohen for his valuable suggestions.

I also would like to thank all my family for motivation, patience, and support that made this work possible.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The current understanding of the world comes from the space-time observations of various phenomena (which are, of course, explained by models of varying complexity, e.g., Kepler's laws, Newton's laws, Murphy's laws, etc.). The phenomena may be expressed as a function (possibly a finite or infinite-dimensional vector function) over the space-time. The observations of that function are taken at some points in space-time and then the function is determined given those observations. In many cases the observations can not be physically taken at the points in space-time, but rather are gathered as averages over small (or not-so-small regions).

This thesis is concerned about ways to determine the underlying function (model) when the observations are integrals or averages over some regions in space-time (or just in space). A particularly challenging problem is when the regions are of irregular form (not simple geometric objects like spheres, cylinders, or boxes). Those types of regions are most common in applications when the data is gathered for administrative, political, geographic, or agricultural regions. A list of examples includes:

- the yearly per capita income by county;
- the weekly number of cases of a reportable disease by state;
- the number of trees by forest subdivision;
- the amount of crop by field.

In each case one can imagine that the quantity of interest could be well-modeled by a smoothly varying function of time and space. The reported data is simply the average value of the function (with respect to some measure) over a region in space-time. In order to estimate (predict) the value of this function one should know the dependence structure of the underlying stochastic process. The existence and form of spatial-temporal dependencies is also an important question. In Chapter 2 we propose a method to estimate the covariance function (or variogram) from the integrals of a stationary stochastic process. The method poses the problem as a set of integral equations which are then solved via least squares. To solve the equations efficiently in the case of an isotropic covariance function in two dimensions we obtain a closed form expression for the kernel functions (the functions that are convolved with the covariance function in the integral equations) in Chapter 3.

We discuss two approaches to predict a space-time process given its integrals; a simple-to-implement kernel type method and a statistically-motivated best linear unbiased predictor (BLUP or kriging) method. The latter method requires the knowledge of the trend and the covariance function of the process. Chapter 4 describes both kernel and kriging type methods for prediction of a space-time process given its integrals.

Having a predicted surface in space-time we discuss ways to present it in the form of an animation, i.e., as a set of images shown in a rapid sequence. This type of visualization is a natural way to present a space-time process, because it has both space (location on the screen) and time (image number) components.

Finally we apply the above methods to visualize data on mumps disease in the United States. Here we consider an animation of a function of three dimensions; two dimensions are represented by space and the remaining one by time. Although showing a set of images in a rapid sequence (the animation) is not a new concept, the use of this method is new in statistics and, in particular, in epidemiology.

With this we finish the cycle from the real-world problem of aggregate data, to the theoretical investigations, and then back to the reality of the mumps dataset.

The following sections describe the work in greater detail. First we consider estimation of a covariance function given integrals of a stochastic process. The estimation requires an efficient way to compute some kernel functions and this is the topic of the next chapter. Using an estimate of the covariance function we can perform best linear unbiased prediction (BLUP) from the integrals of our stochastic process. We also describe an alternative kernel smoothing type prediction method. Finally, we apply the methodology to visualize the mumps data in the United States.

1.1 Estimating a Covariance Function from Integrals of a Stochastic Process

Consider integrals $z_i = \int_{A_i} f(x)dx$ over sets $A_i \subset A \subset R^d$ of a zero-mean stationary process f.

In Chapter 2 we are interested in the covariance function $\gamma(x_1 - x_2)$ of the process f (the estimate will be used to predict the process f in Chapter 4). The estimate of γ can be obtained by solving appropriate integral equations. The following example shows how the estimation of covariance function can be formulated in terms of the solution to integral equations.

Example 1: Let $d = 1$, $A_1 = [-1, 0],\ A_2 = [0, 1]$. Let $f(x)$ be a zero mean stationary process on $[-1, 1]$ with the symmetric covariance function $\gamma(l) = \gamma(-l)$ and $z_1 = \int_{A_1} f(x)dx,\ z_2 = \int_{A_2} f(x)dx$ be two observations. Then

$$\begin{aligned} E(z_i z_j) &= E\left(\int_{A_i}\int_{A_j} f(u)f(v)dudv\right) \\ &= \int_{A_i}\int_{A_j} \gamma(u,v)dudv \end{aligned}$$

so that the products $z_i z_j$ approximate the appropriate integral of the covariance function. As the covariance function is just a function of $|u - v|$ ($\gamma(u, v) = \gamma(|u - v|)$) we can simplify the integral even further. Take $i = 1, j = 2$, then

$$\int_{A_1}\int_{A_2} \gamma(u,v)dudv = \int_{-1}^{0}\int_{0}^{1} \gamma(|u-v|)dudv = \int_0^\infty W_{A_1,A_2}(l)\gamma(l)dl$$

where the kernel function

$$W_{A_1,A_2}(l) = \begin{cases} l & \text{if} \quad 0 \le l \le 1 \\ 2-l & \text{if} \quad 1 \le l \le 2 \\ 0 & \text{otherwise} \end{cases}$$

We can estimate the covariance functon by trying to find a fuction γ that satisfies the integral equations $z_i z_j = \int_0^\infty W_{A_i,A_j}(l)\gamma(l)dl,\ i,j = 1,2.$

A general statistical perspective on solving integral equations can be found in O'Sullivan (1986). An efficient algorithm to estimate γ is developed here for the case of an isotropic covariance function (isotropic means that $\gamma(x_1 - x_2)$ is a function of only $\|x_1 - x_2\|$), $d = 2$, and the A_i's being polygons. Related work on the estimation of the covariance function from point observations can be found in, e.g., Delfner (1976), Kitanidis (1983), Cressie (1985), Marshall and Mardia (1985). Surprisingly, the estimation of the covariance function from integral observations has not been investigated.

The assumption of stationarity of $f(x)$ could be replaced by the existence of the variogram.

Definition 1.1.1 *Let $f(x)$ be a stochastic process such that the quantity $v(t) = \mathrm{E}((f(x) - f(x+t))^2)$ is finite and does not depend on x. Then the function $v(t)$ is called the variogram of the process $f(x)$.*

The variogram could be estimated using equations similar to those derived for the covariance function (see Appendix A.1).

1.2 Deriving the Kernel W

In Chapter 3 we present an algorithm to compute a quantity (see Example 1)

$$W_{AB}(l) = \int_{u \in A, v \in B, \|u-v\|=l} dudv \tag{1.1}$$

for any two finite regions A and B in R^2 with a piecewise linear boundary.

The kernel W_{AB} is needed to estimate the covariance function from the integrals of a stochastic process over the sets A and B (Chapter 2). The covariance function, in its own turn, will be used in prediction of the values of the process (Chapter 4). The kernels W may also be used to generate a random sample of integrals of a stochastic process with any specified isotropic covariance function.

Informally, the kernel W_{AB} is the amount of the movements of a rigid stick so that one end of the stick remains in the region A while the other end remains in the region B. This analogy is suggested the usage of integral geometry framework in Chapter 3 (see, e.g., Bonnesen and Frenchel (1987), Grünbaum (1967), Santalo (1976), Appendix A.2).

The important property of the problem is the shape of the integration regions. The solution is given for the polygonal regions and the idea is to express the integrals over the regions by way of the integrals over the boundary (see Theorem 3.5.4).

1.3 Prediction From Aggregate Quantities

In Chapter 4 we are interested in determining a function $f(x)$ given its integrals $z_i = \int_{A_i} f(x)dx$ for the purpose of animation (see, e.g, Eddy and Mockus (1993a), (1993b)), where A_i's partition a finite set $A \subset R^d$. Modeling the function as a stochastic process and having an estimate of the dependence structure of the process (see Chapter 2) we can perform the best linear unbiased prediction of the process. In Chapter 4 we review existing approaches and introduce new techniques to determine a function given its integrals.

The determination of a function f given several of its functionals $g_i(f)$ is a well studied problem in applied mathematics and statistics. It is called an ill-posed inverse problem. It is an ill-posed problem because there usually are several solutions to the problem unless we severely restrict the space of desired solutions. It is an inverse problem because we have to reconstruct the function from the values of some functionals. For a statistical perspective on solving an ill-posed inverse problem see O'Sullivan (1986).

The most common variation of this problem is when the functionals g_i are values of the function f at points x_i, i.e., $g_i(f) = f(x_i)$. In the statistical approach to the same problem the function f is random and/or the observations include a random error. There are several distinct, well-investigated, ways to determine the function f. Let the function f be nonrandom and the observations $g_i(f) = f(x_i) + \epsilon$, where ϵ is a random error. Then we can estimate $\hat{f}(x)$ that best fits the data and has additional desired properties. Depending on the fitting criteria and on the desired properties of the fitted function we may end up with various kernel type methods, orthogonal function series methods, or spline methods. If the function f is assumed to be random and observations are $g_i(f) = f(x_i)$ then the standard approach is to use one of best linear prediction (or kriging) methods (see, e.g. Cressie (1991)). A comparison between Kriging and spline prediction from point observations in the one-dimensional case can be found in Laslett (1994). The conclusions are that the general cross validation splines (see, e.g., Whaba (1990)) in most cases tend to oversmooth increasing mean squared error. Kriging from meteorological point observations in space-time is described in Handcock and Wallis (1994).

When the observation functionals g_i are integrals of the function f, i.e.,

$$g_i(f) = \int K_i(x) f(x) dx$$

where $K_i(x)$ are known functions, we have to solve those integral equations to determine the function f. This problem is also well investigated and most of the above mentioned methods (except the kernel methods) were widely used to solve it.

A kernel type smoothing method is described in Eddy and Mockus (1994). A similar problem of estimating population density was considered by Tobler (1979). That paper contains a bibliography from the field of applied geography. The interpolation proposed by Tobler is a numeric solution of a Dirichlet's equation with somewhat arbitrary boundary constraints. The method is designed to produce an interpolant under geographic constraints in the form of lakes, mountains, and deserts. Unfortunately the interpolant is not simple to compute and lacks statistical justification.

We describe two interpolation methods. The simplest method to use (kernel type method) (see Section 4.4) is similar to the kernel smoothing methods. We are not aware of anyone using kernel methods to interpolate from aggregate data. A more sophisticated method (see Section 4.2) is based on the assumption that the function of interest is a stochastic space-time process and we construct a best linear unbiased predictor of such process. We will refer to this method as a kriging type method because the best linear unbiased prediction in more than one dimension is often referred to as kriging. While the kernel type method has little justification (except for its simplicity) it produces results (the animation) similar to the kriging type method. The smoothing parameter of the kernel method could be qualitatively assessed from the estimate of the covariance function. Strong long-range correlations would imply that more smoothing is desirable, while weak long-range correlations would imply that less smoothing is needed; the predicted surface in the later case looks more like a step function.

1.4 Applications in Visualizing Mumps Data

Visualization of complex models or data can provide useful insights that are difficult or impossible to detect in other ways. In Chapter 5 an animation of a function of three dimensions is considered; two dimensions represent space and the remaining one time. Although showing a set of images in a rapid sequence (the animation) is not a new concept, the use of this method is new in statistics and, in particular, in epidemiology. There is a substantial amount of work trying to show a high dimensional function on a two dimensional static display (see, for example, Tufte (1983)). Another, more recent approach, is interactive graphics (see, for example, Cleveland and McGill (1988)). In this case some relatively simple projections from a higher dimensional space onto a two-dimensional display are performed and the viewer can interactively change the projection, rotating or scaling the displayed object. An automatic change of projection, a “Grand Tour,” is another possibility (see Asimov (1985)). All those techniques are not very useful for the large amounts of data involved in space-time processes. In epidemiology a standard visualization technique is a static disease map. It was first well documented by Snow for the 1748-1754 cholera epidemics in London (see, e.g., Cliff and Haggett (1992) pp.4-11). Any epidemic, as well as many other processes, changes both in space and time. An animation is an improvement of the static map as it enables us to perceive the change in time visually. Section 5.4 contains a brief discussion on how an animation is produced.

The space-time process is estimated from data that has a particular form. In epidemiology that data usually represent counts of the disease cases in some administrative regions and over some time periods. To estimate the intensity of the disease one has to estimate a function from its integrals over the mentioned administrative regions and time periods.

Small area estimation methods (see, e.g., Ghosh and Rao (1994)) could be appropriate to prepare the data for animations of mumps for individual states or counties. The animation of the whole continental United States does not require such fine details, and, to the contrary, the mumps incidence with spatial precision the size of a county in the considered animation (in the imaginary situation if we had monthly incidence rates for the counties) would look highly discontinuous and would be difficult to understand visually. The main reason is the extremely nonuniform distribution of the population, for example, 54 percent of the population live in 155 most populated counties (5 percent of the total number of counties) and the area of those most populated counties is a negligible percent of the area of the continental US. In our animation those counties look like little specks, yet that is where virtually all mumps cases are concentrated. To show the qualitative behavior (that could be comprehended by viewing the animation) of the spread of mumps on the scale of the United States we need to use substantial amount of smoothing.

A VHS videotape recorded in NTSC standard (used in the United States and Japan) that comes with this thesis contains the animations described in Chapter 5.

Chapter 2

Estimating a covariance function given integrals

2.1 Introduction

We have integrals $z_i = \int_{A_i} f(x)dx$ over sets $A_i \subset A \subset R^d$ of a zero-mean stationary Gaussian process f on A.

In this Chapter we are interested in the covariance function $\gamma(x_1 - x_2)$ of the process f (the estimate will be used to predict the process f in Chapter 4). The estimate of γ can be obtained by solving appropriate integral equations (see Example 1).

The dimensionality d of the process $f(x)$ is important because in the case of a two-dimensional process we provide an efficient implementation of the estimation algorithm; it is difficult to estimate in higher dimensions for general regions A_i. In applications we are frequently interested in three-dimensional space-time processes, but the regions A have irregular boundary only in two spatial dimensions (for example, borders of an administrative or political region). If the integrals are given for spatial regions over a fixed time period those regions in three dimensions (including time) are prisms and this particular form simplifies the estimation process.

The assumption of stationarity of f could be replaced by the existence of a variogram. The variogram could be estimated using equations similar to those derived for the covariance function (see Appendix A.1). A rigorous treatment of generalizations of spatial processes for which the prediction is possible given only one realization can be found in, e.g., Matheron (1973). Those general processes are called Intrinsic Random Functions (IRF) and they possess an analog of a covariance function - a generalized covariance function. The variogram is a simple case of a generalized covariance function.

In Section 2.2 three types of integral equations are obtained: for a covariance function, for a spectral density, and for some (to be defined later in Equation (2.5)) distribution function G. Those equations are then used to estimate the covariance function, the spectral density, or the function G.

In Section 2.3 ways to solve the resulting integral equations are considered.

Section 2.3.3 considers the sufficient conditions for the asymptotic consistency of the proposed estimator.

2.2 Estimation Problem

Let f be a stationary zero-mean Gaussian process on a set $A \subset R^d$ having unknown covariance function γ. Let $z_i = \int_{A_i} f(x)dx$, where $A_i \subset A,\ i = 1, \dots, N$. Then the vector $(z_1, \dots, z_N)$, where $z_i = \int_{A_i} f(x)dx,\ A_i \subset A$, has a multivariate Normal distribution with expected value 0 and covariance matrix

$$\begin{aligned}&\mathrm{Cov}(z_1, \dots, z_N) = \\ &\begin{pmatrix} \int_{A_1}\int_{A_1}\gamma(u,v)dudv & \cdots & \int_{A_1}\int_{A_N}\gamma(u,v)dudv \\ \vdots & \ddots & \vdots \\ \cdots & \cdots & \int_{A_N}\int_{A_N}\gamma(u,v)dudv \end{pmatrix}.\end{aligned} \tag{2.1}$$

The products z_iz_j approximate the integrals $\int_{A_i}\int_{A_j}\gamma(u-v)dudv$ (for a stationary process $\gamma(u,v)$ is a function of only $u-v$), and γ can be estimated by solving an inverse problem for γ:

$$z_iz_j = \int_{A_i}\int_{A_j}\hat{\gamma}(u-v)dudv,\ i,j = 1, \dots, N. \tag{2.2}$$

If the A_i's are regularly spaced, the appropriate z_iz_j could be averaged, reducing the number of equations. In the general case, it is not clear how to reduce the total number of $N*(N+1)/2$ equations.

The solution to an inverse problem is, in general, not a positive definite function, but it can be projected onto the space of nonnegative definite functions. To perform the projection we can do a d-dimensional Fourier transform of the estimate obtained by solving the above-stated inverse problem (2.2) to get the function

$$g(s) = \int \hat{\gamma}(u-v)^T e^{-i(u-v)^T s} d(u-v),$$

where s is an d-dimensional vector and $(u-v)^T s$ is a scalar product.

The estimate g will, in general, be negative in some regions. Taking $g^+ = \max(g, 0)$ we can perform an inverse transform

$$\tilde{\gamma}(u-v) = C \int g^+(s) e^{i(u-v)^T s} ds,$$

where C is an appropriate constant. The function $\tilde{\gamma}$ will be nonnegative definite.

Another approach could be to express γ as a Fourier transform of g and then solve the inverse problem for g. This way we will need to estimate a positive function g instead of a positive definite function γ. The inverse problem is:

$$\begin{aligned} z_i z_j &= \int_{A_i} \int_{A_j} \gamma(u-v) du dv \\ &= \int_{A_i} \int_{A_j} \int g(s) e^{i(u-v)^T s} ds du dv \\ &= \int g(s) V_{A_i A_j}(s) ds, \end{aligned} \tag{2.3}$$

where $V_{A_i A_j}(s) = \int_{A_i} \int_{A_j} e^{i(u-v)^T s} du dv$.

The solution $\hat{g}(s)$ has to be a nonnegative function to ensure nonnegative definiteness of the estimated covariance function $\hat{\gamma}(u-v)) = C \int \hat{g}(s) e^{i(u-v)s} ds$.

For an isotropic covariance function (i.e. $\gamma(u, v) = \gamma(\|u-v\|)$, where $\|\cdot\|$ is Euclidean distance) Equation (2.2) can be rewritten as:

$$\begin{aligned} z_i z_j &= \int_{A_i} \int_{A_j} \gamma(u-v) du dv \\ &= \int_0^\infty W_{A_i A_j}(l) \gamma(l) dl, \end{aligned} \tag{2.4}$$

where $l = \|u-v\|$ and

$$W_{A_i A_j}(l) = \int_{u,v: u \in A_i, v \in A_j, \|u-v\| = l} du dv$$

The functions $W_{A_i A_j}(l)$'s are nonzero only on intervals

$$l \in \left(\inf_{u \in A_i, v \in A_j} \|u-v\|, \sup_{u \in A_i, v \in A_j} \|u-v\| \right).$$

The geometric problem is to obtain the weight function $W_{A_i A_j}(l)$. The functions $W_{A_i A_j}(l)$ are closely related to the kinematic measure $M(l, A_i, A_j)$ (see Appendix A.2) of all movements of an oriented segment of length l such that one end of the segment is in the set A_i and the other end is in the set A_j. In Chapter 3 we obtain an efficient algorithm to calculate the kernels $W_{A_iA_j}$ and $V_{A_iA_j}$ when the regions A_i are in R^2.

Not any positive definite function is an isotropic covariance function in more than one dimensions. A general form of an isotropic correlation function can be found in, e.g., Matèrn (1986). If a correlation function r of a d-dimensional stationary random process is continuous at the origin then there exists a d-dimensional random variable X which has r as characteristic function. The distribution function of that random variable is called the spectral distribution function of the process. For isotropic correlation function $r(l) = E(e^{iu^TX}) = E(e^{il\|X\|Y})$, where $t = \|X\|$ and Y is a random variable independent of $\|X\|$ and uniformly distributed on a unit sphere on R^d. The density function and the characteristic function for Y is

$$f_Y(y) = \frac{(1-y^2)^{(d-3)/2}}{B\left(\frac{d-1}{2}, \frac{1}{2}\right)}, \quad -1 \leq y \leq 1$$

$$\phi_Y(l) = \frac{d-2}{2}! \left(\frac{2}{l}\right)^{\frac{d-2}{2}} J_{\frac{d-2}{2}}(l)$$

where B is the Beta-distribution an J is the Bessel function of the first kind. Taking expectation conditional on $\|X\|$ and then taking expectation with respect to the distribution G of $\|X\|$ we get

$$r(l) = G(0) + \int_0^\infty \phi_Y(ls)dG(s). \tag{2.5}$$

hence the isotropic correlation function is a scale mixture of Bessel functions and the estimation of such function could be done by estimating the function G appearing Equation (2.5). Reformulating problem (2.4) in terms of the function G we get

$$\begin{aligned} z_iz_j &= \int_0^\infty W_{A_iA_j}(l)\gamma(l)dl \\ &= \int_0^\infty W_{A_iA_j}(l)\sigma\left(G(0) + \int_0^\infty \phi_Y(ls)dG(s)\right)dl \\ &= \int_0^\infty Q_{A_iA_j}(s)dG(s) \end{aligned} \tag{2.6}$$

where $Q_{A_iA_j}(s) = \sigma\left(\int_0^\infty W_{A_iA_j}(l)\phi_Y(ls)dl + S_{A_i}S_{A_j}G(0)\right)$, S_{A_i} is the area of A_i, and $\sigma = \gamma(0)$.

Several parametric correlation models for continuous stationary stochastic processes on the plane can be found in the literature. Whittle (1954) considered a spatial AR process generated by two-dimensional Laplace equation

$$\frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} - \phi f(x,y) = \epsilon(x,y),$$

where ϕ is positive and $\epsilon(x, y)$ is a two-dimensional white noise. Vecchia (1985) considered classes of two-dimensional spatial processes with rational spectral density and gave computational formulas for calculating correlations. Jones and Vecchia (1993) consider fitting spatial ARMA processes that are solutions to stochastic differential equation:

$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} - \phi_0\right) \dots \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} - \phi_p\right) f(x, y) = \\ \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} - \theta_0\right) \dots \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} - \theta_q\right) \epsilon(x, y)$$

All those models could be estimated using the methods to solve the inverse problems (2.4) or (2.3) as described in the next section. We will only consider isotropic covariance function so the arguments to the covariance function or the spectral density are scalar.

2.3 Inverse Problem

In this section the integral equations (2.4), (2.3), and (2.6) are solved. As was indicated in Section 2.2 most of the methods that solve integral equations would produce a nonpositive definite estimate for γ (unless we are just estimating a parameter of a family of positive definite functions) if we choose to solve Equation (2.4). The solution to Equation (2.3) needs to be nonnegative and that is a much simpler condition than nonnegative-definiteness.

In general it may be reasonable to solve integral equations in the space domain also. The actual methods used to solve integral equations and the basic properties of the stochastic process should influence the decision in which domain to solve the integral equations. As an example let the process f be nearly white noise. Then the covariance function would decrease dramatically at the origin, complicating its estimation. The spectral density would be close to a constant function, which is an easy function to approximate. If we consider the opposite example, when there exist long distance positive dependencies in the process f, then the covariance function would decrease smoothly and the spectral density would have spikes. This latter case is likely to be easier to solve in the space domain.

Equation (2.6) is more difficult to solve than the other two equations. The advantage is that the approximate solution would be a well defined isotropic covariance function.

2.3.1 Parametrization

Although we are doing "nonparametric estimation", we have to represent γ and g in some finite parametrization in order to solve the integral equations numerically. An example of the parametrization, $g(s)$ could be a step function $g(s) = \sum_{p=0}^{M-1} g_p I_{s_p<s<s_{p+1}}$, where I is an indicator function. We could also choose some spline function or a function series to approximate γ and g.

Denote the parameter vector $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_M)$. The integral equations in this parametrization become:

$$\begin{aligned} z_i z_j &= \int W_{ij}(l)\gamma(l,\boldsymbol{\eta})dl, \\ z_i z_j &= \int V_{ij}(t)g(s,\boldsymbol{\eta})ds, \\ z_i z_j &= \int Q_{ij}(t)dG(s,\boldsymbol{\eta}). \end{aligned}$$

where $W_{ij}(l) = W_{A_iA_j}(l)$, $V_{ij}(l) = V_{A_iA_j}(l)$ and $Q_{ij}(l) = Q_{A_iA_j}(l)$.

Now we could look for the weighted least squares solution to those equations, i.e, for

$$\hat{\boldsymbol{\eta}} = \arg\min_{\boldsymbol{\eta}} \sum_{i,j} C_{ij}\left(z_i z_j - \int W_{ij}(l)\gamma(l,\boldsymbol{\eta})dl\right)^2, \tag{2.7}$$

$$\hat{\boldsymbol{\eta}} = \arg\min_{\boldsymbol{\eta}} \sum_{i,j} C_{ij}\left(z_i z_j - \int V_{ij}(s)g(s,\boldsymbol{\eta})ds\right)^2, \tag{2.8}$$

$$\hat{\boldsymbol{\eta}} = \arg\min_{\boldsymbol{\eta}} \sum_{i,j} C_{ij}\left(z_i z_j - \int Q_{ij}(s)dG(s,\boldsymbol{\eta})\right)^2. \tag{2.9}$$

where C_{ij} are weights that could be inversely proportional to the variance of $z_i z_j$. In that case and when the area S_{A_i} of the regions A_i gets small, the C_{ij} are proportional to $\left(\frac{1}{S_{A_i}S_{A_j}}\right)^2$. In practice we can perform the least squares iteratively; first find $\hat{\boldsymbol{\eta}}$ using $C_{ij} = \left(\frac{1}{S_{A_i}S_{A_j}}\right)^2$, then set $C_{ij} = \frac{1}{(\int W_{ij}(l)\gamma(l,\eta)dl)^2}$ and perform least squares again. The C_{ij} are updated in this way until η stops changing from iteration to iteration. In practice, having the weights $C_{ij} = \left(\frac{1}{S_{A_i}S_{A_j}}\right)^2$ (as opposed to having $C_{ij} = 1$) provided more reliable convergence of the numerical optimization methods in finding the optimal solution.

The refinement of the least squares procedures could be an MLE (Maximum Likelihood Estimate). If we assume the process f to be a stationary zero mean isotropic Gaussian process with the covariance function $\gamma(l,\boldsymbol{\eta})$ plus a fixed trend $\mu(\eta,u)$, then the vector $\{z_i - \int_{A_i}\mu(\eta,u)du\},\ i =$

$1, \ldots, N$ would be have multivariate Gaussian distribution with zero mean and the covariance function given by Equation (2.1). Maximizing the likelihood of the observations $\{z_i\}$ we can obtain the optimal value of $\boldsymbol{\eta}$.

2.3.2 Step Function Approximation

Let $0 = l_0 < l_1 < l_2 < \ldots < l_{M-1} < l_M = \infty$. Let $\gamma(l, \boldsymbol{\eta}) = \sum_{i=0}^{M-1} \eta_i I_{[l_i, l_{i+1})}(l)$ be an approximation to the covariance function $\gamma(l)$. Let $W_{ij}^k = \int_{l_k}^{l_{k+1}} W_{ij}(l)dl$. Then the least squares solution

$$\hat{\boldsymbol{\eta}} = \arg\min_{\boldsymbol{\eta}} \sum_{i,j} C_{ij} \left(z_i z_j - \sum_k W_{ij}^k \eta_k \right)^2 \tag{2.10}$$

can be obtained by solving a system of $N(N+1)/2$ linear equations with M unknowns under constraints that $\boldsymbol{\eta}$ is positive. Unfortunately, the solution will not be a positive definite function. It will not be a reasonable covariance function because if a covariance function is continuous at zero it is continuous everywhere (see, e.g., Cramer and Leadbetter (1967)). The piecewise constant approximation given in Equation (2.10) should be interpreted as an approximation of the integrals of the true covariance function, i.e., $\eta_k(l_{k+1} - l_k) \approx \int_{l_k}^{l_{k+1}} \gamma(l)dl$.

Splines of the first or higher order (step function is a zero order spline) could be used as an approximation of the covariance function. The usefulness of that approach is not clear. The approximation for γ is not likely to be a positive definite function even if we use higher order splines, while the least squares equations are likely to become more complicated.

Let $0 = s_0 < s_1 < s_2 < \ldots < s_{M-1} < s_M = \infty$. Let $g(s, \boldsymbol{\eta}) = \sum_{i=0}^{M-1} \eta_i I_{[s_i, s_{i+1})}(s)$ be an approximation to the spectral density function $g(s)$. Let $V_{ij}^k = \int_{s_k}^{s_{k+1}} V_{ij}(s)ds$. Then the least squares solution

$$\hat{\boldsymbol{\eta}} = \arg\min_{\boldsymbol{\eta}} \sum_{i,j} C_{ij} \left(z_i z_j - \sum_k V_{ij}^k \eta_k \right)^2 \tag{2.11}$$

$$\eta_k \geq 0, \; k = 1, \ldots, M-1 \tag{2.12}$$

can be obtained by solving a system of $N(N+1)/2$ linear equations with M unknowns under constraints that $\boldsymbol{\eta}$ is positive.

Let

$$G(s, \boldsymbol{\eta}) = \sum_{i=0}^{M} \eta_i I_{[s_i, \infty)}(s)$$

be an estimate for the mixing measure G of the scale for the Bessel functions involved in Equation (2.5). Let

$$\begin{aligned}\gamma(l, \boldsymbol{\eta}) &= \int_0^\infty \phi_Y(ls)dG(s, \boldsymbol{\eta}) \\ &= \sum_{i=1}^{M} (\eta_i - \eta_{i-1})\phi_Y(ls_i)\end{aligned}$$

Then we could find least squares solution for $\boldsymbol{\eta}$ by

$$\hat{\boldsymbol{\eta}} = \arg\min_{\boldsymbol{\eta}} \sum_{i,j} C_{ij} \left(z_i z_j - \int W_{ij}(l) \sum_{i=1}^{M} (\eta_i - \eta_{i-1})\phi_Y(ls_i)dl \right)^2 \tag{2.13}$$

or by

$$\hat{\boldsymbol{\eta}} = \arg\min_{\boldsymbol{\eta}} \sum_{i,j} C_{ij} \left(z_i z_j - \sum_{i=1}^{M} Q_{ij}(s_i)(\eta_i - \eta_{i-1}) \right)^2 \tag{2.14}$$

2.3.3 Consistency of the Step Estimators

We will consider asymptotics in terms of the size of the regions A_i getting smaller as well as the size of the region A getting larger. The former is to obtain $\gamma(l)$ when l is small and the latter to obtain $\gamma(l)$ when l is large. Appropriate theorems with proofs are given in Appendix A.4.

2.4 Example

To illustrate the behavior of the covariances between regions we consider a simple parametric case with $\gamma(t) = \sigma e^{\alpha t}$. The three regions A_1, A_2, A_3 partition a 30×30 square region A (see Figure 2.1). In this example we expect the covariance $\text{Cov}(z_1, z_2)$ to be bigger than the covariance $\text{Cov}(z_1, z_3)$ despite the fact that the regions A_2 and A_3 have the same area.

Figure 2.2 shows covariances between all possible pairs of regions for different values of α using covariance function $\gamma(t) = e^{\alpha t}$. As expected, the covariance $\text{Cov}(z_1, z_1)$ is the biggest due to the largest area of A_1. The covariances $\text{Cov}(z_2, z_2)$ and $\text{Cov}(z_3, z_3)$ are identical because the regions A_2 and A_3 are of the same shape and size.

It is interesting to compare plots for the covariances $\text{Cov}(z_1, z_3)$ and $\text{Cov}(z_2, z_3)$. For the values of α close to zero the covariance $\text{Cov}(z_2, z_3)$ is smaller than the covariance $\text{Cov}(z_1, z_3)$ because the region A_1 is larger than the region A_2 and when α is close to zero the correlations decrease slowly with distance, i.e., the larger distance between the regions A_1 and A_3 (than between

Figure 2.1: The three regions A_1, A_2, A_3

A_2 and A_3) does not affect the covariance too much. For the values of α far from zero we have $\mathrm{Cov}(z_1, z_3) < \mathrm{Cov}(z_2, z_3)$ because the regions A_2 and A_3 have a long common border (they are "close"), while the regions A_1 and A_3 have only one point where they meet.

The surface of the sum of squares in Equation (2.7) as a function of σ and α are plotted in Figure 2.3 using $z_i z_j = \mathrm{Cov}(z_i, z_j)$, covariance function $\gamma(t) = e^{-t}$, and weights $C_{ij} = \left(\frac{1}{S_{A_i} S_{A_j}}\right)^2$.

The sum of squares using exact covariances looks like a Rosenbrock function and is difficult to minimize numerically. This kind of surface is a result of "numerically unfriendly" parametrization of the covariance function.

The Rosenbrock function is often used to evaluate the performance of numerical optimization algorithms an is given by following equation:

$$F(x, y) = 100(y - x^2)^2 + (1 - x)^2$$

It has banana shaped level lines.

2.5 Summary

In this chapter the problem of estimating a covariance function of a stationary isotropic process when the available data are integrals of this process was considered. Examples of integral data

Figure 2.2: Covariances as functions of α

include but are not limited to:

- Reports of disease counts over administrative (political) regions.
- Reports of various kinds of census data over census regions.
- Reports on crop yield over different fields.

The estimation of the covariance function from point observations was investigated before, but the estimation from integral observations was not. In the case of point observations the nonparametric form of the covariance function is estimated from the averages of products of observation pairs with a fixed distance apart. When the data are integrals this approach is not applicable.

The proposed solution entails constructing a system of integral equations and then solving by least squares. The three types of integral equations were given by Equations (2.4), (2.3), and (2.6). The kernel functions for those integral equations are obtained in the next chapter. Equation (2.4) was used to estimate the covariance function directly, Equation (2.3) was used to estimate the spectral density, and Equation (2.6) was used to estimate the mixing measure for the scale family of Bessel functions (the mixture of those Bessel functions represents the covariance function of interest).

Figure 2.3: The surface of the sum of squares

We proposed to solve those equations using the least squares approach (alternative, likelihood based approach was mentioned at the end of Section 2.3.1). Estimation of a parametric form of the covariance function (e.g., $\gamma(l) = \sigma e^{-\alpha l}$) can be performed by directly minimizing the sum of squares with respect to the parameter (e.g., (σ, α)).

We considered a nonparametric estimation of the covariance function (i.e., when the number of parameters is asymptotically infinite). We considered approximating the covariance function in Equation (2.7), the spectral density in Equation (2.8), the measure G in Equation (2.9). Simple and computable approximations for all three cases were given in Section 2.3.2. The particular form of step function approximation for the covariance function and for the spectral density (Equations (2.10) and (2.11)) can lead to an estimator that is not necessarily a valid isotropic covariance function in R^2 (although the estimator is asymptotically consistent), while the last two estimators given in Equations (2.13) and (2.14) always represent a valid isotropic covariance function.

A set of conditions for the consistency of the estimator (2.10) were given in Appendix A.4. A simple example was then given to illustrate how the covariances between integrals over regions depend on the covariance function of the underlying stochastic process.

A simple but useful additional result of this chapter is the ability to use the integral equations in opposite direction: to obtain explicit covariances between integrals of a stochastic process when a covariance function is known. Those covariances between integrals of the process can be used to generate the integrals of the process directly.

Chapter 3

Geometric considerations

3.1 Introduction

In this chapter we obtain the kernel W for Equation (2.4). We present an algorithm to compute a quantity

$$W_{AB}(l) = \int_{u \in A, v \in B, \|u-v\|=l} dudv$$

for any two finite regions A and B in R^2 with a piecewise linear boundary. The main result of the chapter is Theorem 3.5.4, which shows relationship between some efficiently computable kinematic measure and the quantity $W_{AB}(l)$.

The kernel W_{AB} is needed to estimate the covariance function from the integrals of a stochastic process over the sets A and B (see Chapter 2). The covariance function, in its own turn, will be used in prediction of the values of the process (see Chapter 4). The kernels W may also be used to generate a random sample of integrals of a stochastic process with any specified isotropic covariance function.

The basic idea of obtaining the kernel W is to replace the two-dimensional double integral

$$\int_A \int_B \gamma(u-v)dudv$$

with the one dimensional integral

$$\int_0^\infty W_{AB}(l)\gamma(l)dl,$$

(also, see Example 1).

Informally, the kernel W_{AB} is the amount of the movements of a rigid stick so that one end of the stick remains in the region A while the other end remains in the region B.

To solve the stated problem we use the terminology and basic facts from integral geometry (see, e.g, Appendix A.2, Santalo 1976, Bonnesen and Frenchel 1987).

The important property of the problem is the shape of the integration regions. The solution is given for the polygonal regions and the idea is to express the integrals over the regions by way of the integrals over the boundary (Theorem 3.5.4).

The kernel $W_{AB}(l)$ is related to the kinematic measure $M(l, A, B)$ (it is proportional to M as a function of A, B up to a linear function of l; for more details see Appendix A.2) of all movements of an oriented segment of length l such that one end of the segment is in the set A and the other end is in the set B. This relation leads to the investigation of the properties of the measure M. Another measure of interest is M_b, the kinematic measure of all movements of an oriented segment of length l intersecting two line segments. We obtain the measure M_b in a closed form and express the measure M through the measure M_b to find the kernel W (see Theorem 3.5.4).

In Sections 3.2 we introduce definitions used later in this chapter and obtain elementary properties of the measures M and M_b. In Section 3.3 a special case of the measure M when both ends of the segment are within the same set is considered. Then, in Section 3.4, a closed form for the elementary boundary measure M_b is obtained. In Section 3.5 all results are put together to get an easily computable expression for W.

In Section 3.6 extension of the results for general regions and for higher dimensions is considered. Then an alternative solution to the original problem is described (Section 3.7). Finally some examples are presented.

3.2 The Kinematic Measure of a Segment with its Endpoints Within Two Sets

To simplify the presentation we will introduce the following notation. Let $M(l, A, B)$ be the kinematic measure (see Appendix A.2) of the movements of an oriented segment K of length l so that K has one end in the interior of A and the other end in the interior of B.

Definition 3.2.1 *Let $M_b(l, A, B)$ be the kinematic measure of the movements of an oriented segment K of length l so that K has nonempty intersection with sets A and B.*

General kinematic measures of the movements of a fixed size oriented segment K will be denoted as $m(K : \ldots)$, where $\ldots$ explicitly specify the restrictions on the possible movements.

Proposition 3.2.2 *Let $C_{A_iA_j}$ be the convex hull of $A_i \cup A_j$, $A_i \cap A_j = \emptyset$ and $M(l, A_i, A_j)$ be equal to the measure of all movements of an oriented line segment K of length l with one end in the set A_i and other end in the set A_j.*

Then

$$\begin{aligned} M(l, A_i, A_j) = M(l, C_{A_iA_j}, C_{A_iA_j}) \\ -M(l, C_{A_iA_j} \setminus A_i, C_{A_iA_j} \setminus A_i) \\ -M(l, C_{A_iA_j} \setminus A_j, C_{A_iA_j} \setminus A_j) \\ +M(l, C_{A_iA_j} \setminus (A_j \cup A_i), C_{A_iA_j} \setminus (A_j \cup A_i)) \end{aligned}$$

Proof:

To simplify the notation let $A = A_i, B = A_j, C = C_{A_iA_j}, D = C \setminus (B \cup A)$. Using Equations (A.4) and (A.5) and biadditivity property of the double integral we get:

$$\begin{aligned} M(l, C, C) &= \frac{1}{l} W_{C,C}(l) \\ &= \frac{1}{l} \left(W_{A,C}(l) + W_{B\cup D,C}(l) \right) \\ &= \frac{1}{l} \left(W_{A,B}(l) + W_{A,A\cup D}(l) + W_{B\cup D,B\cup D}(l) + W_{B\cup D,A}(l) \right) \\ &= \frac{1}{l} \left(W_{A,B}(l) + W_{A\cup D,A\cup D}(l) - W_{D,A\cup D}(l) + W_{B\cup D,B\cup D}(l) + W_{B,A}(l) + W_{D,A}(l) \right) \\ &= \frac{1}{l} \left(2W_{A,B}(l) + W_{A\cup D,A\cup D}(l) + W_{B\cup D,B\cup D}(l) - W_{D,D}(l) \right) \\ &= M(l, A, B) + M(l, A \cup D, A \cup D) + M(l, B \cup D, B \cup D) - M(l, D, D) \end{aligned}$$

□

This proposition implies that it is sufficient to calculate $M(l, X, X)$ (the kinematic measure of all movements of the oriented segment of length l such that both ends of the segment are within some general set X) to obtain $M(l, A_i, A_j)$. Note that the usefulness of this proposition might be limited as the set $X = C_{A_iA_j} \setminus (A_j \cup A_i)$ can be unconnected if sets A_j, A_i are not convex (see Figure 3.1).

3.3 The Kinematic Measure of a Segment Inside a Convex Set

Given any convex set K_0 of area F_0 and perimeter L_0 the kinematic measure of all movements of an oriented segment K of length l such that $K \cap K_0 \neq \emptyset$ is

$$m(K; K \cap K_0 \neq \emptyset) = 2\pi F_0 + 2lL_0 \tag{3.1}$$

Figure 3.1: A_i, A_j, and their convex hull $C_{A_iA_j}$

(see Santalo pp. 90). We are interested in the measure of all movements of a segment such that $K \subset K_0$ ($M(l, K_0, K_0)$). For any convex set K_0 the intersection of the segment K with the border ∂K_0 can have zero, one, two, or infinitely many points, i.e. $\mathrm{card}(K \cap \partial K_0) = 0, 1, 2, \infty$. It is easy to prove that for any convex set K_0, $m(K; \mathrm{card}(K \cap \partial K_0) = \infty) = 0$. The cases when $\mathrm{card}(K \cap \partial K_0) = \infty$ will be disregarded in the future discussion as that does not change the measure of interest.

Proposition 3.3.1 *The measure $M(l, K_0, K_0)$ of all movements of an oriented segment K such that K is inside a convex set K_0 can be written as follows:*

$$M(l, K_0, K_0) = m(K; K \subset K_0) = m(K; K \cap K_0 \neq \emptyset) - M_1 - M_2, \tag{3.2}$$

where $M_1 = m(K; \mathrm{card}(K \cap \partial K_0) = 1)$, $M_2 = m(K; \mathrm{card}(K \cap \partial K_0) = 2)$.

Proof: This equation is a consequence of the fact that if the segment K intersects the convex set K_0 it may have zero, one, or two points in common with the border of K_0 (the case of infinitely many points has kinematic measure zero). If it intersects K_0 and does not intersect ∂K_0 then it is inside the set K_0. $\Box$

Let K_0 be a convex polygon. The measure $M(l, K_0, K_0)$ is obtained by Santalo (pp. 91-92) for the particular case when the segment K can not intersect two nonadjacent sides of the polygon, i.e.

when $M_2 = 0$.

The border ∂K_0 of a polygon is a union of n segments s_i with lengths l_i. For each s_i, $M_b(l, s_i, s_i) = m(K; K \cap s_i \neq \emptyset) = 4ll_i$ according to Eq. (3.1) as the area of s_i is zero and the perimeter is $2l_i$. Hence summing $m(K; K \cap s_i \neq \emptyset)$ over all border segments s_i we get $\sum_i 4ll_i = M_1 + 2M_2$ as K can intersect at most two border segments at a time and when it does the measure is counted for both of them.

Proposition 3.3.2 *The measure $M(l, K_0, K_0)$ of all movements of a segment K such that K is inside a convex set K_0 of area F_0 and perimeter L_0 is*

$$M(l, K_0, K_0) = 2\pi F_0 - 2lL_0 + M_2, \tag{3.3}$$

Proof:

$$\begin{aligned} M(l, K_0, K_0) &= m(K; K \cap K_0 \neq \emptyset) - M_1 - M_2 \\ &= 2\pi F_0 + 2lL_0 - (M_1 + 2M_2) + M_2 \\ &= 2\pi F_0 + 2lL_0 - 4lL_0 + M_2 \\ &= 2\pi F_0 - 2lL_0 + M_2, \end{aligned}$$

The first equality is from Proposition (3.3.1), the second equality is obtained from Equation (3.1), the third equality is is an application of Poincare's formula (see Santalo pp. 111). □

For a convex polygon K_0 with the boundary consisting of segments s_i we get $M(l, K_0, K_0) = 2\pi F_0 - 2l \sum \|s_i\| + M_2$ and the quantity M_2 can be easily expressed using measure M_b

$$\begin{aligned} M_2 &= m(K; \operatorname{card}(K \cap \partial K_0) = 2) \\ &= \frac{1}{2} \sum_{s_i \neq s_j,\, s_i, s_j \in \partial K_0} M_b(l, s_i, s_j) \end{aligned}$$

Noting that $\frac{1}{2} M_b(l, s_i, s_i) = 2l\|s_i\|$, we can rewrite Equation (3.3) for the case of a convex polygon

$$M(l, K_0, K_0) = 2\pi F_0 + \frac{1}{2} \sum_{s_i, s_j \in \partial K_0} (-1)^{\delta_{ij}} M_b(l, s_i, s_j), \tag{3.4}$$

where $\delta_{ij} = \begin{cases} 1 & \text{if} \quad i = j, \\ 0 & \text{if} \quad i \neq j. \end{cases}$

Figure 3.2: Two segments forming an angle α intersected by a third one

3.4 The Kinematic Measure of a Segment Intersecting Two Other Segments

Let two segments OA and OB with lengths l_1 and l_2 have one end (O) in common and an angle $\alpha > 0$ between them be intersected by a segment K of length l (see Figure 3.2). The measure $M_b(l, AO, OB) = m(K; K \cap OA \neq \emptyset \text{ and } K \cap OB \neq \emptyset)$. In this section we will use shorter notation $M_{AO,OB} = M_b(l, AO, OB)$. This measure for other possible positions of the segments AB and BC (see Figure 3.3) can be obtained using the measure for the case shown in Figure 3.2. In general case (Figure 3.3)

$$M_{AB,CD} = M_{AB,OC} + M_{AB,OD} = M_{OB,OC} - M_{OA,OC} + M_{OB,OD} - M_{OA,OD}$$

Denote the rays originating at points O, O, A, B and going in the direction of vectors $\overrightarrow{OB}, \overrightarrow{OA}, \overrightarrow{OA}, \overrightarrow{OB}$ accordingly as $R(OB), R(OA), R(A\infty), R(B\infty)$. From Figure 3.4 we get

$$M_{AO,OB} = M_{R(OB),R(OA)} - M_{R(OB),R(A\infty)} - M_{R(B\infty),R(OA)} + M_{R(B\infty),R(A\infty)}.$$

The measures on the right hand side of the equation are (for derivation see Appendix A.3):

$$M_{R(OB),R(OA)} = \frac{l^2\left(1 + (\pi - \alpha)\cot(\alpha)\right)}{2} \tag{3.5}$$

Figure 3.3: General case of two segments AB and CD.

Figure 3.4: $M_{AO,OB} = M_{R(OB),R(OA)} - M_{R(OB),R(A\infty)} - M_{R(B\infty),R(OA)} + M_{R(B\infty),R(A\infty)}$

$$\begin{aligned}
&M_{R(B\infty),R(OA)} = && (3.6)\\
&\quad \int_{\psi_0}^{\psi_1} \sin(\phi)\left(l + \frac{l_2\,\sin(\alpha)}{\sin(-\phi+\alpha)}\right)\left(\frac{l\sin(\phi-\alpha)}{\sin(\alpha)} - l_2\right)d\phi\\
&M_{R(B\infty),R(A\infty)} = && (3.7)\\
&\quad \int_{\psi_{10}}^{\psi_{11}} \sin(\phi)\left(l + \frac{l_2\,\sin(\alpha)}{\sin(-\phi+\alpha)}\right)\left(\frac{l\sin(\phi-\alpha)}{\sin(\alpha)} - l_2\right)d\phi\\
&+ \quad \int_{\psi_{20}}^{\psi_{21}} \sin(\phi)\left(l + \frac{l_1\,\sin(\alpha)}{\sin(-\phi+\alpha)}\right)\left(\frac{l\sin(\phi-\alpha)}{\sin(\alpha)} - l_1\right)d\phi,
\end{aligned}$$

where

$$\begin{aligned}
\psi_0 &= \arcsin\left(l_2/l\sin(\alpha)\right) + \alpha\\
\psi_1 &= \min(\pi, \pi - \arcsin\left(l_2/l\sin(\alpha)\right) + \alpha)\\
\psi_{10} &= \arcsin\left(l_2/l\sin(\alpha)\right) + \alpha\\
\psi_{20} &= \arcsin\left(l_1/l\sin(\alpha)\right) + \alpha\\
\psi_{11} &= \arcsin\left(l_2/\sqrt{l_1^2 + l_2^2 - 2l_1l_2\cos(\alpha)}\sin(\alpha)\right) + \alpha\\
\psi_{21} &= \arcsin\left(l_1/\sqrt{l_1^2 + l_2^2 - 2l_1l_2\cos(\alpha)}\sin(\alpha)\right) + \alpha,
\end{aligned}$$

Formula (3.6) is valid when $l_2\sin(\alpha) < l$, otherwise $M_{R(B\infty),R(OA)} = 0$. Formula (3.7) is valid when $l_1^2+l_2^2-2l_1l_2\cos(\alpha) < l_2$, otherwise $M_{R(B\infty),R(A\infty)} = M_{R(B\infty),R(OA)}$ or $M_{R(B\infty),R(A\infty)} = M_{R(OB),R(A\infty)}$.

The integrals appearing in Equations (3.6,3.7) are available in closed form, for example, Equation (3.6) is:

$$\begin{aligned}
&\int_{\psi_0}^{\psi_1} \sin(\phi)\left(l + \frac{l_2\,\sin(\alpha)}{\sin(-\phi+\alpha)}\right)\left(-\frac{l\sin(-\phi+\alpha)}{\sin(\alpha)} - l_2\right)d\phi =\\
&\qquad 2ll_2(1+\cos(\phi)) - l^2\left(\frac{\sin(\phi)^2}{2} + \cot(\alpha)\frac{\sin(2\phi)}{4}\right)\\
&+ \left(l_2^2\sin(2\alpha) + l^2\cot(\alpha)\right)\frac{\phi}{2} + l_2^2\sin(\alpha)^2\ln\left(\sin(\phi-\alpha)\right)\Big|_{\psi_0}^{\psi_1} \qquad (3.8)
\end{aligned}$$

The case of parallel segments ($\alpha = 0$) has to be handled separately. Assume the situation as shown in Figure 3.5. Given a fixed orientation of K (it has angle ϕ with l_1) the measure of all non-rotational movements of K such that $K \cap l_1 \neq \emptyset$ and $K \cap l_2 \neq \emptyset$ is equal to the area of the shaded parallelogram (see Figure 3.5). Let the distance between the segments l_1 and l_2 be d and the

Figure 3.5: Two parallel segments l_1, l_2 intersected by the segment K.

shift be s as shown in Figure 3.5. Then

$$\begin{aligned} M_{l_1,l_2} &= 2\int_0^{\pi} \max(0, l\sin(\phi) - d)\max(0, b-a)d\phi, \\ a &= \max(0, s - d\tan(\pi/2 - \phi)), \\ b &= \min(l_1, s + l_2 - d\tan(\pi/2-\phi)), \end{aligned} \tag{3.9}$$

where l, l_1, l_2 are lengths of the segments K, l_1, l_2.

3.5 Relation of M and M_b for General Polygonal Sets

In this section a signed measure M^* derived from the measure M_b is introduced. Then a theorem which gives the relationship between the measure M and the measure M^* is stated and proved. As the measure M^* is directly related to the measure M_b (which, in its own turn, is easily computable) we can compute the measure M and hence the kernels W and V.

To get a relationship between the measures M and M_b in a general case we need a couple of definitions.

Definition 3.5.1 *Let $\overrightarrow{AB}, \overrightarrow{CD}$ be two nonzero length, oriented line segments and $L_{AB} \cap L_{CD} \notin (\overrightarrow{AB} \cup \overrightarrow{CD}) \setminus (A \cup B \cup C \cup D)$, where L_{AB}, L_{CD} are lines defined by the segments. Then $\overrightarrow{AB}, \overrightarrow{CD}$*

are called separated.

Definition 3.5.2 *Let $\overrightarrow{AB}, \overrightarrow{CD}$ be separated segments. Let K be an oriented segment of length l intersecting $\overrightarrow{AB}$ and $\overrightarrow{CD}$. Define the signed measure*

$$M^*(l, \overrightarrow{AB}, \overrightarrow{CD}) = \\ \operatorname{sign}\left(\|AC\| + \|BD\| - \|AD\| - \|BC\|\right) M_b(l, \overrightarrow{AB}, \overrightarrow{CD}),$$

where M_b is the kinematic measure of the movements of a segment K intersecting segments AB and CD.

Because the kinematic measure is defined in terms of integrals it is not difficult to show the following:

Lemma 3.5.3 *M^* is finitely biadditive on line segments, i.e.*

$$M^*(l, A \cup C, B) = M^*(l, A, B) + M^*(l, C, B) - M^*(l, A \cap C, B) \tag{3.10}$$

$$M^*(l, A, B \cup C) = M^*(l, A, B) + M^*(l, A, C) - M^*(l, A, B \cap C) \tag{3.11}$$

for any three pairwise separated line segments A, B, and C.

Using biadditivity of $M^*(l, P, Q)$ we can extend the definition to an arbitrary pair of segments and to an arbitrary pair of sets of segments. Given a finite set of segments $s \in S$ we can construct a finite set of separated segments $s^* \in S^*$, so that $\cup_{s \in S} s = \cup_{s^* \in S^*} s^*$ where the union is taken considering the segments as point subsets of R^2. The construction is as follows: for each $s \in S$ draw a line L_s defined by s. Consider all intersection points between every line and each non-parallel (to the line) segment. Those points will divide the segments s into smaller segments s^*. By construction every pair s_1^*, s_2^* is separated, as the set of lines created from the segments $s \in S$ is the same as the set of lines from the segments $s^* \in S^*$.

Let a polygonal set P in R^2 be any set having a boundary consisting of a finite number of separated line segments and equal to the closure of its opening, i.e. $P = \overline{P \setminus \partial P}$. Let B_P be a set of counter-clockwise oriented boundary segments $\vec{p}$ so that for each $\vec{p} \in B_P$ the interior of the set P is on left side of $\vec{p}$. More formally, let $\overrightarrow{AB} = \vec{p}$ and for any $\epsilon > 0$ there exist $\delta > 0$, such that vector $A + t\vec{p} + s\vec{o} \in P \setminus \partial P$, where $t \in (\epsilon, 1-\epsilon)$, $s \in (0, \delta)$, and $\vec{o}$ is a unit vector having angle $\pi/2$ with $\vec{p}$ in counter-clockwise direction ($(\vec{v}, \vec{o}) = 0$ for the scalar product (angle is $\pi/2$ or $-\pi/2$), and $\vec{v} \times \vec{o} < 0$ for the vector product in a right handed coordinate system).

Theorem 3.5.4 *Let P and Q be two polygonal sets. Let B_P, B_Q be sets of clockwise (counter-clockwise) oriented boundary segments of P and Q. Then*

$$\int_P \int_Q g(\|x_p - x_q\|) dx_p dx_q = \int_0^\infty W(l) g(l) l dl, \tag{3.12}$$

where g is any function such that integral on the left exists and

$$W(l) = 2\pi F_{P\cap Q} + \frac{1}{2} \sum_{p\in B_P} \sum_{q\in B_Q} M^*(l,p,q) = 2\pi F_{P\cap Q} + \frac{1}{2} M^*(l, B_P, B_Q), \tag{3.13}$$

where $F_{P\cap Q}$ is the area of $P \cap Q$.

To prove this statement we will have to obtain several intermediate results. First we will obtain a similar result for two non-overlapping triangles, then we will extend it to the case of an arbitrary union of non-intersecting triangles, and finally we will prove the general case.

Let AB and CD be two arbitrary separated line segments from the counter-clockwise oriented boundaries B_P, B_Q of two non-intersecting triangles P and Q. Segment AB divides a plane in two half-planes so that the triangle P is only in one half-plane. Using this fact when considering a segment K intersecting AB and CD, we can answer following two questions:

1. Can K have its end inside P?

2. Can K have its end inside Q?

Following table summarizes answers for different possible orientations of the segments. Illustration is provided in Figure 3.6.

Type	AB	OC	Answer 1	Answer 2	Sign of M^*
0	$\overrightarrow{BA}$	$\overrightarrow{CD}$	yes	yes	$+$
1	$\overrightarrow{AB}$	$\overrightarrow{CD}$	no	yes	$-$
2	$\overrightarrow{BA}$	$\overrightarrow{DC}$	yes	no	$-$
3	$\overrightarrow{AB}$	$\overrightarrow{DC}$	no	no	$+$

Table 3.1: List of the types of the line segment pairs, their orientation, answers to questions 1 and 2, and the sign of the signed measure for the given orientation

We can classify all pairs (p, q) ($p \in B_P$ and $q \in B_Q$) into four types described in the table. It should be noted that for pairs of type 1 and 3 the segment K has exactly two intersection points with triangle P (except when the intersection is a vertex of P). This implies that the movements

Figure 3.6: Possible pairwise orientations of AB and CD.

of K when it intersects pairs p, q of type 0 contain all other movements of K (only the movements such that $K \cap P \neq \emptyset$ and $K \cap Q \neq \emptyset$ are considered). Similarly, the movements when K intersects pairs of type 3 are contained in the movements of type 2 and also in the movements of type 3. Let T_i be a set of movements of K so that it intersects at least one pair of type i. It is easy to see that $T_1 \cap T_2 = T_3$. We are interested in the kinematic measure of the movements $T_0 \ (T_1 \cup T_2)$, i.e. $M(l, P, Q) = m(K : T_0 \ (T_1 \cup T_2)$. Using set additivity of kinematic measure and the sign of M^* from Table 3.1

$$\begin{aligned}
& m(K : T_0 \ (T_1 \cup T_2)) \\
&= m(K : T_0) - (m(K : T_1 \cup T_2) \\
&= m(K : T_0) - m(K : T_1) - m(K : T_2) + m(K : T_3) \\
&= \sum_{p,q:(p,q) \text{ type } 0} M^*(l, p, q) + \sum_{p,q:(p,q) \text{ type } 1} M^*(l, p, q) \\
&\quad + \sum_{p,q:(p,q) \text{ type } 2} M^*(l, p, q) + \sum_{p,q:(p,q) \text{ type } 3} M^*(l, p, q) \\
&= \sum_{p \in B_P, q \in B_Q} M^*(K, p, q),
\end{aligned}$$

which leads to the following lemma

Lemma 3.5.5 *Let P and Q be two non-overlapping triangles in R^2. The kinematic measure of an oriented segment of length l with one end in P and the other end in Q is*

$$M(l,P,Q) = \sum_{p \in B_P} \sum_{q \in B_Q} M^*(l,p,q) = M^*(l,B_P,B_Q),$$

where B_P and B_Q are oriented boundary of P and Q accordingly.

Any polygonal set P is a union of finitely many disjoint triangles $P = \cup_{i=1}^n P_i$ that could be obtained triangulating each polygon in P. Let $P = \cup P_i$ and $Q = \cup Q_i$ be two non-intersecting polygonal sets, where P_i and Q_i represent a partition of P and Q into triangles. Using set biadditivity of M

$$\begin{aligned} M(l,P,Q) &= \sum_{i,j} M(l,P_i,Q_j) = \\ &= \sum_{i,j} \sum_{p_i \in B_{P_i}, q_j \in B_{Q_j}} M^*(l,p_i,q_j) \\ &= \sum_{p \in B_P, q \in B_Q} M^*(l,p,q). \end{aligned} \tag{3.14}$$

The last step can be obtained by noting that if any two triangles from the same polygonal set share a boundary segment, i.e. un-oriented segment p_i is the same as un-oriented segment p_j, then those segments must have opposite orientations (boundary is always oriented in one direction, say, counter-clockwise) and so for any other segment q separated from p_i we have $M^*(l,p_i,q) + M^*(l,p_j,q) = M^*(l,p_i,q) - M^*(l,p_i,q) = 0$.

To prove the general case of Theorem 3.5.4 consider two, possibly intersecting polygonal sets P, Q. Let $PP = P \setminus Q$, $PQ = P \cap Q$, and $QQ = Q \setminus P$. PP, PQ, and QQ are non-intersecting polygonal sets. Using Equations (A.5) and (A.4),

$$\begin{aligned} W_{P,Q}(l) &= W_{PP,PQ}(l) + W_{PP,QQ}(l) + W_{PQ,QQ}(l) + W_{PQ,PQ}(l) \\ &= (M(l,PP,PQ) + M(l,PP,QQ) + M(l,PQ,QQ))l/2 \\ &\quad + M(l,PQ,PQ)l \end{aligned} \tag{3.15}$$

To find $M(l,PQ,PQ)$ consider a partition of PQ into triangles PQ_i. Then

$$\begin{aligned} M(l,PQ,PQ) &= \frac{1}{2} \sum_{PQ_i \in PQ, PQ_j \in PQ,\, i \neq j} M(l,PQ_i,PQ_j) \\ &\quad + \sum_{PQ_i \in PQ} M(l,PQ_i,PQ_i) \end{aligned} \tag{3.16}$$

The quantity $M(l, PQ_i, PQ_i)$ can be obtained using Equation (3.4), using the fact that $M^*(l, p, q) = M_b(l, p, q)$ for any two different oriented boundary segments p, q of a convex set, and using the fact that $M^*(l, p, p) = -M_b(l, p, p)$ for any oriented segment p.

$$M(l, PQ_i, PQ_i) \quad = \quad 2\pi F_{PQ_i} + \frac{1}{2} \sum_{p \in B_{PQ_i}, q \in B_{PQ_i}} M^*(l, p, q)$$

where F_{PQ_i} is the area of PQ_i. Hence, Equation (3.16) becomes:

$$\begin{aligned} M(l, PQ, PQ) &= \sum_{PQ_i \in PQ} 2\pi F_{PQ_i} + \frac{1}{2} \sum_{PQ_i \in PQ, PQ_j \in PQ} M^*(l, B_{PQ_i}, B_{PQ_j}) \\ &= 2\pi F_{PQ} + \frac{1}{2} M^*(l, B_{PQ}, B_{PQ}) \end{aligned}$$

as all terms corresponding to the common boundary between the triangles PQ_i cancel each other in the sum.

From Equation (3.15) and Lemma 3.5.5,

$$\begin{aligned} W_{P,Q}(l) &= \frac{l}{2}(M^*(l, B_{PP}, B_{PQ}) + M^*(l, B_{PP}, B_{QQ}) \\ &\quad + M^*(l, B_{PQ}, B_{QQ}) + M^*(l, B_{PQ}, B_{PQ})) \\ &= \frac{l}{2}(M^*(l, B_P, B_{PQ}) + M^*(l, B_{PP}, B_{QQ}) + M^*(l, B_{PQ}, B_{QQ})) \\ &= \frac{l}{2}(M^*(l, B_P, B_{PQ}) + M^*(l, B_P, B_{QQ})) \\ &= \frac{l}{2} M^*(l, B_P, B_Q). \end{aligned}$$

The second and third equalities are consequence of the fact that PQ and PP share a part of their boundary, and the fourth equality uses the fact that PQ and QQ share a part of their boundary, in particular, $(\partial PP \cup \partial PQ) \setminus (\partial PP \cap \partial PQ) = \partial P$. This completes the proof of Theorem 3.5.4.

3.6 Nonpolygonal Areas and Extension to Higher Dimensions

A trivial way to proceed in the case of nonpolygonal areas A_i would be to approximate the areas of interest by polygons and use described algorithms. To do a precise approximation may require polygons having a lot of vertices and that could make the problem too difficult.

Let P, Q be two finite connected sets, and B_P, B_Q their clockwise oriented boundary. Formally we could consider a contour integral

$$M^*(l, B_P, B_Q) = \oint_{B_P} \oint_{B_Q} M^*(l, dp, dq),$$

Figure 3.7: Two prisms in R^3.

as a generalization of the sum in the Equation (3.13). The usefulness of such generalization in practical applications is questionable.

In higher dimensions the problem of finding M_{A_i,A_j} is much more complicated unless the regions A_i have a special form. An irregular planar region considered over a fixed time period is a prism in three dimensional space (position on the plane and in time). It is not difficult to extend the current two dimensional results to three dimensions when the regions A_i are prisms. Let A and B be orthogonal prisms in R^3 with the bases A_2 and B_2 on the same plane p (see Figure 3.7). Let h be a common height of the prisms and let $u = (u_1, u_2, u_3)$, $v = (v_1, v_2, v_3)$ be a Cartesian coordinate system such that the first two components denote coordinates in the plane p. Then the quantity

$$\begin{aligned}
W_{A,B}(l) &= \int_{u\in A, v\in B,\ \|u-v\|=l} du dv \\
&= \int_0^1 du_3 dv_3 \times \\
&\quad \int_{(u_1,u_2)\in A_2,(v_1,v2)\in B_2,\ \sum_{i=1,2}(u_i-v_i)^2=l^2-(u_3-v_3)^2} du_1 du_2 dv_1 dv_2 \\
&\quad \int_0^1 W^*_{A_2,B_2}(l^2-(u_3-v_3)^2) du_3 dv_3
\end{aligned} \tag{3.17}$$

where $W^*_{A_2,B_2}$ is the appropriate kinematic measure in R^2. This result generalizes to higher dimensions, we just need to do the one dimensional integral of $W^*_{A_2,B_2}$ in any (higher than two) dimensional Euclidean space to get the appropriate function W. It should be noted that the regions A and B in four or more dimensions can not be any orthogonal prisms; irregular boundary is allowed only for two dimensions and those dimensions must be the same for both regions. The regions must be boxes in the remaining dimensions.

The measure M_{A_i,A_j} is difficult to calculate for a general polytope in R^n. In applications the regions in high dimensions usually have a very specific form (it is not easy to imagine a general polytope in more than three dimensions), so this specific form might lead to a simpler calculation of M_{A_i,A_j}.

3.7 Discrete Approximation

Instead of computing the functions $W_{A_iA_j}(l)$ exactly we could divide the region A into a grid of small squares S_{op} and approximate the integral in Equation (2.2) by the sum over those squares. Equation (2.2) then becomes:

$$\begin{aligned} z_i z_j &= \sum_{\{o,p:S_{op}\in A_i\}} \sum_{\{q,r:S_{qr}\in A_j\}} \gamma(\text{Distance}(S_{op},S_{qr}))\text{Area}(S_{op})\text{Area}(S_{qr}), \\ & \quad i,j=1,\ldots,m. \end{aligned}$$

Using standard filling algorithms for raster graphics we can easily check if a square S_{op} belongs to the region A_i. The number of computations can be prohibitive if we want to improve the approximation. If we take A_i, A_j to contain on the order of 100^2 grid squares then to obtain the integral in Equation (2.2) we have to perform an order of 100^4 calculations. For a twice finer grid, one would require $2^4 = 16$ times more calculations.

3.8 Kernel V

The kernel

$$V_{AB}(s) = \int_A \int_B e^{i(u-v)s} du dv$$

is connected to the kernel $W_{AB}(l) = \int_{u\in A, v\in B, ||u-v||=l} du dv$. If we define

$$W'_{AB}(l) = \begin{cases} W_{AB}(l)/2 & \text{if} \quad l \geq 0 \\ W_{AB}(-l)/2 & \text{if} \quad l < 0 \end{cases}$$

Figure 3.8: The kernels $W_{A_i,A_j}(l)$

Then $V_{AB}(t) = e^{ilt}W'_{AB}(l)$ The function $e^{ilt}W'_{AB}(l)$ seems to be not integrable in a closed form with respect to l. This means that the kernel $V_{AB}(t) = \int e^{ilt}W'_{AB}(l)dl$ has to be obtained numerically. The precise way in which the kernel V is to be obtained depends on the method used to solve integral Equations (2.2, 2.3).

3.9 Example

To illustrate the behaviour of the kernel weight functions we consider a simple example shown in Figure 2.1. The three regions A_1, A_2, A_3 partition a 30×30 square region A. Kernels W for all pairs of regions are shown in Figures 3.8.

The kernels V are real valued as the function W' is symmetric. The plots of kernels V are shown in Figure 3.9. A closeup is provided in Figure 3.10. The function V was calculated by performing the fast Fourier transformation of the values of the function $2W'$ at 243 points equally spaced on the interval $(-30\sqrt{2}, 30\sqrt{2})$. The range for s was interval $2\pi\frac{121}{30\sqrt{2}} \times (-1, 1)$. As the function V is symmetric it was plotted only for positive values of s.

Figure 3.9: The kernels $V_{A_i,A_j}(l)$

Figure 3.10: The closeup on kernels $V_{A_i,A_j}(l)$

3.10 Summary

In this Chapter we present an algorithm to compute a quantity

$$W_{AB}(l) = \int_{u \in A, v \in B, \|u-v\|=l} du dv$$

for any two finite regions A and B with a piecewise linear boundary. This quantity is essential in estimating the covariance function of the stochastic process given its integrals over the regions A as described in Chapter 2. The quantity may also be used to generate a random sample of integrals of a stochastic process with any specified covariance structure.

To obtain the quantity W_{AB} we first notice that it is related to the kinematic measure $M(l, A, B)$ of all movements of an oriented segment of length l with one end in set A and the other end in set B (Equations (A.4) and (A.5)). In Section 3.2 we show that this measure can be expressed in terms of the kinematic measure of all movements of an oriented segment with both ends in the same set.

In Section 3.3 we relate the measure $M(l, A, B)$ to the kinematic measure $M_b(l, S_a, S_b)$ (see Definition 3.2.1) of all movements of an oriented segment of length l intersecting two line segments $S_a,\ S_b$. We provide a closed form formula for the measure M_b in terms of l, S_a, and S_b (see Equations (3.6, 3.7, 3.5, 3.9). The closed form expression for integrals involved in those measures is given in Equation (3.8).

The measure M_b can be generalized to a signed measure M^* (see Section 3.5) which depends on the orientations of the segments S_a and S_b. The measure M^* is then extended to the finite unions of line segments.

The boundaries of the regions A and B involved in Equation (1.1) are finite unions of line segments so the measure M^* is defined on them.

As the most important result we find a direct relationship between the quantity $W_{AB}(l)$ and the measure $M^*(l, \partial A, \partial B)$ in Theorem 3.5.4.

Then, in Section 3.6, we extend the results for a general (non-polygonal) regions and to higher than two-dimensional spaces. We conclude with an example.

Chapter 4

Prediction from spatial aggregates

4.1 Introduction

This chapter is about determining a function $f(x)$ given its integrals $z_i = \int_{A_i} f(x)dx$ for the purpose of animation (see, e.g, Eddy and Mockus (1993a), (1993b)), where A_i's partition a finite set $A \subset R^d$.

We consider the particular ill-posed inverse problem of determining a function f described at the beginning of the section with an intent of producing an animation of f. The observed values are $z_i = \int_{A_i} f(x)dx$ where A_i's partition a finite set $A \subset R^d$, so the simplest interpolant for the function f could be a piecewise constant function

$$\hat{f}(x) = \sum_i I_{x \in A_i} \frac{z_i}{\int_{A_i} 1dx} \tag{4.1}$$

Discontinuity of such function can be distracting in animations and needs to be avoided (see Eddy and Mockus (1994)). We consider methods that produce a continuous interpolant.

In Section 4.2 the related work on prediction of the stochastic process is described. Various ways to remove the trend are listed in Section 4.3. Section 4.4 contains description of a kernel type method to interpolate a function from its integrals.

4.2 Best Linear Unbiased Prediction

In this section we first review related work and then introduce motivation for BLUP in Subsection 4.2.1.

In many applications data sets consist of observations $z_1, \ldots, z_n$ taken at corresponding locations $\mathbf{x}_1, \ldots \mathbf{x}_n$. The locations $\mathbf{x}_i$ are usually points in d-dimensional Euclidean space R^d. A fairly common analysis of such data is based on the assumption that it is derived from a stochastic process $F(\mathbf{x}) : \mathbf{x} \in A \subset R^d$. The process F is called a random field. The data are derived from a single realization f of the field F. For the inference to be generally possible one of two assumptions about F are usually made:

1) F is stationary in the wide sense (second order stationary).

2) F is an intrinsic random function of order k (IRF-k) for some integer k (Matheron (1973)).

Under assumption 1) the expected value $E(F(\mathbf{x})) = c$ and does not depend on $\mathbf{x}$. There exist a nonnegative definite covariance function

$$\gamma(\mathbf{u} - \mathbf{v}) = E((F(\mathbf{u}) - c), (F(\mathbf{v}) - c)).$$

Under assumption 2) $E(F(\mathbf{x}))$ is a polynomial in $\mathbf{x}$ of degree k. There exist a generalized covariance function (see, e.g., Matheron (1973)).

When the (generalized) covariance function is known, the best linear unbiased predictor (BLUP or kriging predictor) exists and the expressions for it and its mean squared prediction error are well known. However, in practice the (generalized) covariance function is rarely known; consequently, the parameters of a (generalized) covariance function are first estimated and then used in prediction. The quality of such prediction was considered for a finite sample case by Zimmerman and Cressie (1992), for an asymptotic case by Stein (1991). A Bayesian analysis in predicting spatial functions can be found in, for example, Kitanidis (1986).

4.2.1 The Aggregate Data Problem

Let $F(\mathbf{x})$ be a stochastic process with the index $\mathbf{x} \in A \subset R^d$. Let the observed values z of the single realization f of the process F be

$$z_i = \int I_{A_i}(x) f(t) dt, \tag{4.2}$$

where I_{A_i}'s are indicator functions for disjoint sets $A_i \subset A$.

Let $g(f)$ be a linear operator that is the quantity of interest which would typically be $f(x)$ (value at some point t), or the whole function $f(\cdot)$.

For a prior distribution $\pi(f)$ and a loss function $L(g(f), \hat{g}(\mathbf{z}))$ where $\hat{g}$ is a function of the data $\mathbf{z} = (z_1, \ldots, z_n)$, we can look for $\hat{g}$ that minimizes posterior loss:

$$\hat{g} = \arg\min_{\hat{g}} \int L(g(f), \hat{g}(\mathbf{z}))dP(f|\mathbf{z}),$$

where $P(f|\mathbf{z})$ is the posterior distribution for the realization f.

For the squared error loss ($L(g, \hat{g}) = (g - \hat{g})^2$) we have

$$\begin{aligned} \hat{g} = \arg\min_{\hat{g}} \int (g - \hat{g})^2 dP(f|\mathbf{z}) & \\ = \quad \arg\min_{\hat{g}} E(g^2|\mathbf{z}) + \hat{g}(\hat{g} - 2E(g|\mathbf{z})) & \\ = \quad E(g|\mathbf{z}), & \qquad (4.3) \end{aligned}$$

where $E(g|\mathbf{z})$ is a posterior mean of g given $\mathbf{z}$, and we assumed that $E(g^2|\mathbf{z}) < \infty$. When g is a function, L may be the integrated mean square error loss ($L(g, \hat{g}) = \int (g(s) - \hat{g}(s))^2 ds$), and if $\int_A g(s)ds < \infty$ with probability one (and ds is countably finite), then we can exchange the order of integration and obtain $\hat{g}(s) = E(g(s)|\mathbf{z})$ almost everywhere with respect to the measure ds. Hence, in both cases the distribution of the scalar quantity $g|\mathbf{z}$ or $g(s)|\mathbf{z}$ is of interest.

In case when the joint distribution of $(g(s), \mathbf{z})$ is normal we have $E(g(s)|\mathbf{z}) = \hat{g}(\mathbf{z})$ where $\hat{g}$ is a well known linear function of $\mathbf{z}$ given by the formula for the conditional normal distribution. Let $\mu(s) = E(g(s))$, $\mu_i = E(z_i)$, $c_i(s) = \text{Cov}(g(s), z_i)$, $c_{ij} = \text{Cov}(z_i, z_j)$ calculated a priori. Then the posterior expectation

$$E(g(s)|\mathbf{z}) = \mu(s) - (c_i(s))(c_{ij})^{-1}((\mu_i) - \mathbf{z}). \qquad (4.4)$$

In case when we just know the first two moments of the vector $(g(s), \mathbf{z})$ Equation (4.4) defines best linear unbiased predictor for $g(s)$.

4.2.2 Prediction

Kriging Equation (4.4) involves quantities μ and c that are unknown. The trend $\mu(\cdot)$ is usually assumed to be a constant (see Section 4.3 for other approaches) and we could estimate $c_i(s), c_{ij}$ as described in Chapter 2.

4.3 Estimators of the Trend

In some cases the spatial process has a nonconstant trend. In the case of linear trend we can estimate variogram (see Section A.1) and do best linear unbiased prediction based on the estimated variogram (see, e.g., Cressie 1991). When the trend is a higher order polynomial we could estimate a generalized covariance function of higher order and then do appropriate prediction.

A nonparametric estimate of the trend can be obtained using spatial moving medians (see Section 5.4.3). The trend could then be subtracted from the observed values and the prediction could be done on the residuals.

4.4 Kernel Type Estimator

The usual kernel estimator for $f(x)$ given $z_i = f(x_i)$ is

$$\hat{f}(x) = \frac{\sum_i K(x, x_i) z_i}{\sum_i K(x, x_i)},$$

where $K(x, x_i)$ is a kernel function. When data are aggregate $z_i = \frac{\int_{A_i} f(x) d\mu(x)}{\int_{A_i} d\mu(x)}$ one could define an estimator by analogy

$$\hat{f}(x) = \frac{\sum_i z_i \int_{A_i} K(x, x_i) d\mu(x_i)}{\sum_i \int_{A_i} K(x, x_i) d\mu(x_i)}.$$

An implementation of such method is described in 5.4

Some statistical justification for the kernel smoothing over integrals can be given. Lets consider following version of kriging. Let observations $y_i = \int K(x)(f(x) + W(x))dx$ where K is the kernel function (from a space $\mathcal{K}$ of infinitely differentiable functions with bounded support), f is the process of interest, and W is white noise which is uncorrelated with f. The integral is interpreted as linear random functional on L^2 closure of the space of functions $\mathcal{K}$ (see, e.g., Gelfand and Vilenkin (1964)).

The best linear unbiased predictor of $f(0)$ is defined by the kernel K satisfying the unbiasedness condition $\int K(x)dx = 1$ and the projection property:

$$\text{Cov}\left(f(0) - \int K(x)(f(x) + W(x))dx, \int G(x)(f(x) + W(x))dx\right) = 0$$

for all $G \in \mathcal{K}$ satisfying $\int g(x)dx = 0$. The covariance can be written as

$$\int G(x) \left[\gamma(x) - \int K(y)\gamma(x - y)dy - \sigma K(x)\right] dx = 0,$$

where $\int \sigma K(x)G(x)dx = \int K(x)W(x)dx \int G(x)W(x)dx$. By arbitrariness of the function G the expression in the square brackets is constant. The solution to the system of equations

$$\gamma(x) - \int K(y)\gamma(x - y)dy - \sigma K(x) = C$$
$$\int K(x)dx = 1$$

defines optimal kernel K for the continuous kriging problem. In Stein (1990) an expression for kernel K was found for a particular form of generalized covariance function. It was also shown that asymptotically the kernel predictor was equivalent to the universal kriging predictor. Although the results were obtained for the case of point observations data the asymptotic considerations for the integral data are similar.

4.5 Example

To illustrate the behavior of various predictors we consider a simple example shown in Figure 2.1. The stochastic process f was generated on a regular 30×30 grid using covariance function $\gamma(x) = e^{-0.1\|x\|}$. The realization of the process and the piecewise constant predictor are shown In Figure 4.1.

Figure 4.1: The sample path of f and a piecewise constant estimator.

We obtained optimal values for (α, σ) minimizing Equation (2.7) for the parametric form of covariance function $\gamma(x) = \sigma e^{\alpha\|x\|}$. The observed values $z_i = \int_{A_i} f(x)dx$ and optimal values for (α, σ) are given in Table 4.1.

The predicted surfaces using actual and estimated parameters are in Figure 4.2.

z_1	z_2	z_3
280.98087612230	220.46143366014	99.605256094011

Parameter	α	β
Actual	-0.1	1
Estimated	-0.0361647	0.780573

Table 4.1: The observed data with actual and estimated parameters of the covariance function

Figure 4.2: The predicted f using actual (on the left) and estimated parameters of the covariance function.

4.6 Summary

In this chapter we reviewed existing spatial prediction methods that can be used together with the estimate of the covariance function. Section 4.2 described a general approach for best linear prediction of a continuous spatial process. The method uses the covariance function of the process to calculate the conditional mean of a Gaussian process (with the same first and second moments) given the data. Section 4.2.1 considers specifics of prediction from aggregate data. In Section 4.3 we described various ways to estimate the trend of a spatial process using observations that are integrals of the process. We also present an alternative simple to implement kernel type method in Section 4.4. This method convolves kernel with a function (not with point observation as is usual with standard kernel methods) that is constant over the regions where the data was collected. We provide some informal justification for the use of such a method. Finally, we conclude with a simple example to illustrate the prediction methods.

Chapter 5

Example

5.1 Introduction

Visualization of complex models or data can provide useful insights that are difficult or impossible to detect in other ways. Here an animation of a function of three dimensions is considered; two dimensions represent space and the remaining one time. Although showing a set of images in a rapid sequence (the animation) is not a new concept, the use of this method is new in statistics and, in particular, in epidemiology.

We describe animation of mumps incidence rates in the United states. In Section 5.2 and 5.3 the data is described. Section 5.4 contains description of the process of producing the animation. Section 5.6 contains description of the animations of the mumps recorded on videotape.

5.2 Description of the data

The data on the mumps disease that were collected in the United States from 1957 until 1989 by NNDSS was obtained from the “statlib” library.

- The first dataset consists of the monthly mumps cases for each state in the United States for the period from 1968 until 1988. The data are not available for some months and in some states. There are 33 cases reported with an unidentified month in a year.

- The second dataset has sizes of population for every county as given by 1970 and 1980 censuses.

The monthly state data consists of following records:

State Fips Code	$(year - 1900)$	Month	Number of Cases

The number 13 was used for unidentified month. There are 10342 records. The first date of reporting is January 1968, the last - December 1988. The maximum of the number of cases - 3,298 was reported in Wisconsin in January 1968. The minimum number of reported cases was 1, because the Centers for Disease Control does not as part of normal processing store zeroes in the data. The highest incidence rates of the disease (96 per $100,000$ population) was reported in Iowa on March 1968. The total number of reported cases is 880,240.

5.3 History of mumps in the US

Mumps is a seasonal disease. The peak occurs in early spring, while the lowest incidence rates can be observed in early autumn. As most of the mumps cases are school age children, this seasonal behavior can in part be explained by the school year. Over a longer period mumps had higher incidence rates before state-level vaccination programs started at the end of 1960's. By the end of 1970's these vaccination programs almost completely wiped out the disease, leaving only a few cases per state per month. Vaccination programs were stopped in some states after awhile and strong outbreaks of the disease occurred in 1986-1987 and in 1989, primarily among unvaccinated adolescents and young adults in states without requirements for mumps vaccination. This story is well supported by the graph (Figure 5.1) of the logarithm of incidence rates in California and Wisconsin. A seasonal periodicity can be seen (high in spring and low in autumn) and an outbreak in Wisconsin in the second half of the eighties.

The plot in Figure 5.2 shows behavior of the mumps disease over a longer period of time. The top of the plot shows the percent of population in the reporting states and the bottom shows the logarithm of mumps cases in the reporting states. The top part demonstrates that the adjustment of the bottom plot to account for the cases in non-reporting states is small. The mumps disease had periodic peaks (every three years) and sharply decreased during the vaccination programs of the seventies. There is also a big peak in the 1986 when the large mumps outbreak occurred. This plot was produced using yearly state data. A barely visible dotted line (it is very close to the solid line) on the bottom part of the plot was produced from the monthly state data summing over the seasons of disease (from September until August) to reduce correlations between years. The result is close to the sums over the calendar years. The enlarged version of the two plots is in Figure 5.3

Figure 5.1: Log of the mumps monthly incidence rates versus months from Jan. 1968 to December 1988 for California and Wisconsin

Figure 5.2: Behavior of the mumps disease from 1953 until 1989

Figure 5.3: Behavior of the mumps disease from 1968 until 1989 as given by summing over calendar months (from January until December) in solid line and by summing over the disease season (from September until August) in dotted line

5.4 Animation techniques

Our goal is to estimate and display a smoothly varying scalar function f with a three dimensional argument (x, y, t) as was described in the beginning of Section 5.1.

The following features of an animation should be noted:

1) The time dimension is substantially different from space dimensions. The perception of time and space are quite different. The concept of spatial distribution versus temporal variations at a fixed location are also dissimilar.

2) The eye can not readily distinguish a single pixel from its neighbors.

3) Nonsmooth changes in time are more difficult to detect than similar changes in space.

The first and third properties suggest that interpolation in space could be done independently of interpolation in time.

The second property suggests that spatial interpolation could be done to some subset of the image pixels (preferably a regular grid) and remaining pixels could be filled using simple bi-linear (see Section 5.5.4) interpolation. This would save computation time without degrading the perceived smoothness of the animation.

The next subsection contains description of scaling the animated function (its values and its arguments) to fit into the range of available colors, pixels, and video frames. Then we do temporal interpolation to animate the data and do smoothing to the raw data before animation.

5.4.1 Scaling

Once the function of interest $f(x, y, t)$ is determined, an animation is like a generalization of plotting to three dimensions. To produce an animation one has to scale a four-dimensional object into the graphical device coordinates. Plotting a function of a one dimensional argument only requires scaling of a two-dimensional object (the values of a function and its argument) to fit on the screen or paper.

A function of three dimensions has to be scaled to be shown on a device with finite spatial and temporal resolution. We map the range of the function values into 256 color values. The region where the function is shown is mapped into the 1000×563 pixels of our display window and the time interval of one or two minutes (there are 1800 video frames per minute). These values are limitations of human perception and/or video equipment. According to experiments (see Levkowitz and Herman, 1992) people can clearly distinguish around 120 ordered color values (i.e., they can clearly say which color is “smaller” and which color is “bigger”) on the appropriately chosen color scale. The number of distinct pixels on a television screen is limited by the bandwidth of the display device screen imposing a range of possible values for two dimensions x, y. One or two minutes of technical video is close to becoming boring to anyone other than a subject matter specialist, limiting possible values in the remaining dimension t.

Mumps incidence decreases dramatically from 1968 to 1988. In order to better use the color scale we made a nonlinear transformation of the incidence rates before transforming them linearly into colors. The transformation we chose was to use the empirical distribution function of the data so that the resulting colors would be approximately uniformly distributed over the color scale. We tried using linear and logarithmic scales, but in that case only large variations of incidence rates in the beginning of the period (1968-1978) were detectable, leaving the later period without visible action. In the image processing literature such a transformation of the pixel intensities is called “equalization” (see, for example, Pavlidis, 1982). The actual color scale is displayed at the bottom of the frame with the corresponding incidence rates given just above the colors.

5.4.2 Temporal Interpolation

The entire mumps data set consists of 252 months. NTSC video is displayed at the rate of 30 frames per second (NTSC is the television signal that is used in the United States and Japan). After several experiments we decided that displaying the data at the rate of 20 frames per month was a reasonable compromise between the time required to look at the entire data set and the apparent speed with which changes take place. Thus each month is displayed for two-thirds of one second. If the recording were done so that twenty identical frames were recorded and then the switch were made to the next month's data, the viewer would be distracted by the jumpiness of the resulting images (see Section 5.6.1). Consequently, we chose to interpolate linearly between consecutive months. Precisely, the correctly colored maps for two consecutive months are calculated and then 19 intermediate maps are calculated by linear interpolation in the color scale. This results in substantially smoother appearance.

We considered other types of temporal interpolation (e.g, sinusoidal, trapezoidal, and quadratic), but linear interpolation seemed to be adequate. Other types of interpolation produced additional visual artifacts (like monthly swinging) which interfered with the display of seasonal variations present in the data.

5.4.3 Smoothing Raw Data

Observed data usually contains a substantial amount of noise which, if not removed, can produce a "jumpy" animation which, in turn, could hide interesting features of the animated process.

In time series analysis data are frequently smoothed using running averages or running medians to remove noise. Given a series of observations z_i, the value of the running median at time i is

$$\hat{z}_{i,k} = \mathrm{Median}\{z_j : |j - i| \leq k\},$$

where k is called the size of the running median. Our definition of running median of size k is often referred to as "running median of $2k + 1$", but the latter terminology does not extend to the multidimensional case.

Tobler and Kennedy (1985) used an interpolation from spatial averages. We use spatial (and space-time) medians to smooth the data, not just interpolate missing values. We prefer to use medians (as opposed to averages) because averages are not invariant under the transformation we used.

In the mumps data there are both time and space components, so we could do running medians in time for every region, do moving medians in space for every time moment, or do running-moving medians in time and space together. To define moving medians in space we need to define adjacencies between the locations of observations because the simple (total) ordering by time is no longer present. In our case, regions A_i (states) form a partition of A (the continental US). We define two spatial regions to be adjacent (or 1-adjacent) if they share a common border consisting of more than one point. If there is a region to which they both are adjacent then we call them 2-adjacent. Note that a pair of regions that are 1-adjacent are automatically 2-adjacent. Similarly we can define k-adjacent regions. Given the values z_i for regions A_i we define a moving median of size k at the region A_i as $\hat{z}_{i,k} = \mathrm{Median}\{z_j : A_j$ is k-adjacent to $A_i\}$. The time dimension can be thought of as just another space dimension and then we can apply the moving medians in space and time simultaneously.

We successfully used those techniques to improve the smoothness of animations. We found that moving medians of size one (in space-time) produce a substantial amount of smoothing (see Section 5.6.1).

5.5 Estimation from Spatial Averages

5.5.1 The Problem

The problem of interest is to estimate a function $f(x, y, t)$ (incidence rates of the mumps disease at some location and time moment (x, y, t) given sets $\{A_j\}$ and data $\{z_j\}$. Henceforth we will denote a space-time location as $\mathbf{x} = (x, y, t)$ to simplify the notation. The relationship between f and data is given by following equation

$$z_j = \int_{\mathbf{x} \in A_j} f(\mathbf{x}) dG(\mathbf{x}), \;\; j = 1, \ldots, N$$

where $A_j \subset A \subset R^3$ and $G(\mathbf{x})$ is the population distribution.

Assuming that $f(\mathbf{x})$ is random, one can look for the predictor $\hat{f}$ that minimizes the mean square error (MSE),

$$MSE(\hat{f}(\mathbf{x})) = E((f(\mathbf{x}) - \hat{f}(\mathbf{x}))^2).$$

The function $f(\mathbf{x})$ has to be estimated over some set of $\mathbf{x} \in A \subset R^3$, so the integrated MSE

$$\int_A MSE(\hat{f}(\mathbf{x})) d\mathbf{x},$$

or maximal MSE

$$\sup_{\mathbf{x} \in A} MSE(\hat{f}(\mathbf{x}))$$

could be of interest depending on the problem at hand.

Let $f(\cdot)$ be a zero mean stationary process and $c_i(\mathbf{x}) = E(f(\mathbf{x}) \cdot z_i)$, and let $C = (c_{ij})$ where $c_{ij} = E(z_i \cdot z_j)$. Then the minimum MSE predictor for $f(\mathbf{x})$ would be

$$\hat{f}(\mathbf{x}) = c(\mathbf{x})C^{-1}\mathbf{z}, \tag{5.1}$$

where $c(\mathbf{x}) = (c_i(\mathbf{x})$ is a vector of length N, C^{-1} is the inverse of the $N \times N$ covariance matrix for the $\{z_i\}$, and $\mathbf{z} = (z_i)$ is the observation vector of length N. When the assumption that $f(\cdot)$ is a zero mean process is unreasonable the mean could be estimated taking global or local averages of the observations z_i.

Equation 5.1 has some drawbacks. It requires inversion of the matrix C as well as knowledge of the covariance function of the process to obtain $c_i(\mathbf{x})$ and c_{ij}. In the case of observations at a point ($z_i = f(\mathbf{x}_i)$) there are parametric and nonparametric ways to estimate the covariance function (see, e.g., Cressie, 1991). When data are aggregate, as is in our case, it is still possible to estimate the covariance function (see Chapter 2). Unfortunately, covariance function estimation is difficult so we also considered alternative simpler solutions.

The estimation problem we are considering could be modified so that the interpolation would be done given the values (instead of integrals) of f at some points. This approach is described in next section.

5.5.2 Transforming the Problem

For the interpolation problem when data are values of the function at some points there exist a wide range of fast and simple-to-implement algorithms. A kernel estimator for $f(\mathbf{x})$ given $z_i = f(\mathbf{x}_i)$ is

$$\hat{f}(\mathbf{x}) = \frac{\sum_i K(\mathbf{x}, \mathbf{x}_i) z_i}{\sum_i K(\mathbf{x}, \mathbf{x}_i)},$$

where $K(\mathbf{x}, \mathbf{x}_i)$ is a kernel function. When data are aggregate

$$z_i = \frac{\int_{A_i} f(\mathbf{x}) dG(\mathbf{x})}{\int_{A_i} dG(\mathbf{x})}$$

one could define an estimator by analogy

$$\hat{f}(\mathbf{x}) = \frac{\sum_i z_i \int_{A_i} K(\mathbf{x}, \mathbf{x}_i) dG(\mathbf{x}_i)}{\sum_i \int_{A_i} K(\mathbf{x}, \mathbf{x}_i) dG(\mathbf{x}_i)}.$$

To approximate the integrals $\int_{A_i} K(\mathbf{x}, \mathbf{x}_i) dG(\mathbf{x}_i)$ we took a sample of points uniformly distributed within each state A_i (see Figure 5.11). The number of points sampled in each state was taken proportional to the area of the state. We assigned the value to those points to be equal to the incidence rate for the particular state the points are in. In this way we take into account the differing areas and shapes of the states.

These sampled points are used to interpolate a function to every pixel on the map. We used a weighted combination of the function values at the sampled points to obtain the value at all pixels. The weight function $K(\mathbf{x}, \mathbf{x}_i)$ was chosen to be exponential in the squared distance between the sampled point $\mathbf{x}_i$ and the pixel $\mathbf{x}$ where the function was being interpolated.

The method we used to estimate $f(\mathbf{x})$ in the animations can be described as follows:

- Choose a set of points and a set of values for every A_j, $\mathbf{x}_{ij} \in A_j, z_{ij},\ i = 1, \ldots, k_j$.
  - we took the number of points k_j in the region A_j to be proportional to the area of A_j.
  - The points $\mathbf{x}_{ij}$ are distributed in A_j so that they repel each other and the boundary of A_j. A point $\mathbf{x}_{i+1,j}$ is sampled uniformly from the set $A_j \setminus \cup_{k=1}^{i} r_k$, where r_k is a disk with a center at $\mathbf{x}_{kj}$. The radius of r_k depends on the total number of points to be sampled from A_j and on the size of A_j.
  - The values of f at $\mathbf{x}_{ij}$ are assumed constant for each A_j, $z_{ij} = z_j$.
- Use the estimator
$$\hat{f}(\mathbf{x}) = \frac{\sum_i K(\mathbf{x}, \mathbf{x}_{ij}) z_{ij}}{\sum_i K(\mathbf{x}, \mathbf{x}_{ij})}$$
with $K(\mathbf{x}, \mathbf{x}_{ij}) = e^{-\lambda \|\mathbf{x}_{ij} - \mathbf{x}\|^2}$.
- Choose the smoothing parameter λ to provide an acceptable degree of smoothness to the animation.

5.5.3 Best Linear Unbiased Prediction

Here we implement the methods described in Chapter 2 and Chapter 4.

First we prepare the data, then estimate the covariance function, and, finally, perform the prediction.

Preparing the Data

We consider the incidence rate of mumps (number of disease cases divided by the population size) for a particular state to be the aggregate measure of the disease in that state.

Figure 5.4: The approximate boundary of the states

The state is specified as a region in R^2 by a set of its boundary line segments. The boundary was obtained by taking the boundary line segments expressed as a sequence of points (latitude and longitude pairs) from the “maps” database available on statlib (e-mail: statlib@stat.cmu.edu, see also Becker and Wilks (1991)). The boundary was then transformed using Albers Equal Area Projection with 25 and 45 degree base latitudes (see Deetz and Adams (1921)). To speed up further calculations the boundaries were “thinned” to reduce the number of line segment pieces in the boundary without changing the visual appearance of the map (at the resolution of 1001×607 pixels). The resulting boundaries were used in the following calculations. Using different thinning parameter we produced two sets of boundaries shown in Figures 5.4 and 5.5. The approximate boundary shown in Figure 5.4 was used to check the sensitivity of the estimation on small perturbations of shape.

Functions $W_{A,B}(l) = \int_{u \in A, v \in B,\ \|u-v\|=l} dudv$ are plotted for several states in Figure 5.6.

The incidence rates of the mumps (for i’th state and j’th time period) were transformed to make the empirical distribution look more like a normal distribution (see Equation 5.2). The logarithmic transformation is reasonable with the count data (the incidence rates were obtained dividing counts by the population size). To avoid undefined results when transforming zero counts we took a very small power transformation equivalent to logarithm. All following inference (up to a different scale factor) was identical for both, small power and logarithmic transformations.

Figure 5.5: The exact boundary of the states

Figure 5.6: The functions $W_{A,B}(l)$ for several states. The abscissas are in miles, the ordinates have no dimension $\frac{W^{1/3}}{\mathrm{Area}(A)^{1/2}}$

Figure 5.7: The values (points) and the trend $\overline{y_i(\cdot)}$ (line) of $\left(\frac{\text{Cases}_i(j)}{\text{Population}_i(j)}\right)^{1/100}$

$$y_i(j) = \left(\frac{\text{Cases}_i(j)}{\text{Population}_i(j)}\right)^{1/100} \tag{5.2}$$

The values of $y_i(j)$ are plotted in Figure 5.7. The trend $\overline{y_i(\cdot)} = \frac{\sum_k y_i(k)}{\sum_k 1}$ (only non-missing values were averaged) and the histogram of the residuals $y_i(j) - \overline{y_i(\cdot)}$ are in Figure 5.8.

To obtain aggregate quantities over the states we multiply the incidence rate by the area of the state to get the integral of the incidence rate over the state.

$$z_i(j) = \left(y_i(j) - \overline{y_i(\cdot)}\right) \text{Area}(\text{State}_i) C \tag{5.3}$$

Those quantities $z_i(j)$ were used to estimate the covariance function and to predict the process representing the 0.01 power of the mumps incidence rates. The constant C was used to scale the sum of squares into the range appropriate for the numeric optimization methods.

Estimating the Covariance Function

We face several questions while estimating the covariance function. Should we estimate the covariance function:

1. for each month?
2. for the months in the second period starting from January 1975.
3. for the average (median) of the observed covariance matrices.

Figure 5.8: The histogram of the residuals $y_i(j) - \overline{y_i(\cdot)}$

4. in parametric or nonparametric form?

5. in spectral or spatial domain?

We chose to estimate a parametric $\sigma e^{\alpha|x|}$ covariance function in spatial domain. We chose to estimate the covariance function according to 1), 2), 3) and then to compare the results in Table 5.1.

The estimates of the covariance function have following interpretation. Let half-distance be a distance d such that the covariance function of the process f satisfies $\gamma(d)/\gamma(0) = \frac{1}{2}$ (this definition makes sense only for a monotone covariance function). The half-distance is the distance at which the correlations between the values of the process drop to a half. When the half-distance increases so do the dependencies, when the half-distance approaches zero the process is nearly a white noise.

Prediction

The Best Linear Unbiased Predictor for the zero mean process f at a point s is given by following Equation.

$$E(f(s)|\mathbf{z}) = (c_i(s))(c_{ij})^{-1}\mathbf{z}, \tag{5.4}$$

where $c_i(s) = \mathrm{Cov}(f(s), z_i)$, $c_{ij} = \mathrm{Cov}(z_i, z_j)$, and $z_i = \int_{A_i} f(s)ds$.

The matrix c_{ij} is computed only once (using estimated covariance function), and an appropriate submatrix (depending on which states do not report the mumps cases) is inverted for every month. The vector $c_i(s)$ is computed only for the values of s on a regular grid (see Figure 5.11). The process f was predicted only on a regular grid, the time trend (see Equation 5.3) was added, and then interpolation to every time frame and every pixel was performed as described in Section 5.4.2 and Section 5.5.4.

5.5.4 Two Levels of Interpolation

Estimation using exponential weights for each point as described in Section 5.5.2 can be very time consuming. The frame buffer (the device that generates the NTSC video signal) has more than 500$\times$500 pixels. Assuming an average of 10 points in each state where the value of the function is assumed to be given, we have to perform approximately 10^8 distance and exponential weighting calculations for each frame. This is a substantial amount of time even on a fast workstation given that we want to record 252$\times$20 of those frames.

Estimates for the Mean Covariance Matrix

Case	α	σ	Half-distance
All Dataset	-16.5	672.6	126
First Period	-16.6	767.8	125.3
Second Period	-16.1	480.9	129.2

Estimates for the Median Covariance Matrix

Case	α	σ	Half-distance
All Dataset	-17.1	364	121.6
First Period	-17.7	501	117.5
Second Period	-17.5	273	118.8

Estimates for Each Month

Function	α	σ	Half-distance
Median	-18.7	699	111.2
First Quartile	-26.7	535	78.9
Third Quartile	-14.7	996	141.5

Estimates for Each Month in the First Period

Function	α	σ	Half-distance
Median	-17.5	740	118.7
First Quartile	-25.4	582	81.9
Third Quartile	-14.1	996	146.9

Estimates for Each Month in the Second Period

Function	α	σ	Half-distance
Median	-20.4	668	101.9
First Quartile	-28.8	464	72.2
Third Quartile	-16.1	997	129.2

Table 5.1: The estimates of the covariance function. Half-distances are given in miles.

Consequently the weighted estimation was performed only onto a regular grid over the United States (see Figure 5.11). We chose the grid size to be 35 points by 25 points. We then used a bilinear interpolant from the four values at the corners of each of the 34 times 24 rectangles of the regular grid to each pixel within a particular rectangle.

$$f(i,j) = \sum_{k=0,K,l=0,L} f(k,l)\frac{K-|k-i|}{K-0}\frac{L-|l-j|}{L-0},$$

where (i,j) are coordinates of a pixel within the rectangle, K is the width of the rectangle in pixels, and L is the height of the rectangle in pixels. The weights for each pair (regular grid point, sampled point) are computed only once and stored.

5.6 Videotapes

The animations were produced one after another, improving the result at each step. Despite those improvements the first steps are of interest by themselves.

The simplest possible animation is to display the raw data: a constant value of the incidence rates for each state during every month. The result is difficult to understand due to sharp changes between adjacent states, abrupt changes in time, and abundance of unreported cases. This animation creates a desire for a smoother picture in space and in time.

A better looking picture can be produced by smoothing the raw data and estimating missing values. In this case the smoothed data is displayed as being constant across each state and interpolated between months. The interpolation between months removes “jumps” in time. Both linear and sinusoidal interpolations look reasonably good, but with the sinusoidal interpolation more time is spent showing actually observed (as opposed to interpolated) values, while it also produces monthly swinging effect (the picture changes rapidly between the months, and the picture stops changing and seems to be constant in the middle part of the month) that may interfere with the display of seasonal variations. Various approaches are possible to smooth the data in space. From a practical point of view, running medians in space and time (see Section 5.4.3) is a simple method that also fills in the missing values. Although it is an improvement over the first step this animation still has jumps at state boundaries.

The last step was to produce a smooth animation both in time and space. This required use of techniques described in Section 5.5.2 and Section 4.2. The animation that is smooth in time and space turned out to be visually appealing and easier to understand.

Figure 5.9: Raw incidence rates in December 1986

A brief description of the equipment we used to produce those animations can be found in Eddy and Mockus (1993).

5.6.1 Nonsmooth in Space

Nonsmooth in Time

We have used the background color to indicate missing data. The states which are missing seem to "disappear" into the background when there is no data. An initial version of the videotape switched instantaneously from a color to background when there was a missing observation and then back to a color from background when there was data. The abruptness of this scheme was sufficiently jarring that We modified the scheme to "fade" to background. This is actually done by linear interpolation between the particular color and the background color. One frame of this animation is displayed in Figure 5.9.

The time smoothing was performed as described in Section 5.4.2. Simple linear interpolation in time was the first method we used. We tried other interpolation methods but could not detect any improvement.

Figure 5.10: The locations of points used for kernel smoothing

Filling in Missing Data

The number of states not reporting mumps cases increases in the later part of the data. This distracts the viewer from the overall pattern of the disease. We used methods described in Section 5.4.3 to fill in the missing values. To indicate the fact that the value was not reported we used a dotted fill pattern for the particular state. This way it was possible to show the overall predicted pattern of the disease together with information showing which part of the data was actually reported.

Smoothing in Missing Data

In an attempt to reveal the major patterns in the data we used moving medians as described in Section 5.4.3 not only to fill in the missing values but also to smooth the existing values. This resulted in large regions in space and time having roughly the same color.

5.6.2 Smooth in Space and Time

We produced two animations based on smoothing algorithms in Sections 4.2 and 5.5.2 The smoothest animation was produced using independent time and space smoothing. The smoothing in space was done for every month. First we estimated the intensity on a regular 35×25 grid of points (see Figure 5.11) using the algorithm described in Section 5.5.2 and 4.2. The particular set of points $\mathbf{x}_{ij}$ (using the notation of Section 5.5.2) is shown in Figure 5.10.

To obtain estimates for the remaining pixels we used a simple bi-linear interpolation onto a regular grid (see Figure regular.ps) described in Section 5.5.4. As in previous animations we chose

Figure 5.11: The regular grid with state boundaries

to interpolate linearly the 19 intermediate frames between the monthly smoothed maps. Thus the smoothing in space and in time are independent of each other. The single frame corresponding to December 1986 is displayed in Figure 5.12.

5.6.3 Detecting Disease Outbreaks

As an alternative to showing the incidence of the disease we considered inspection of the residuals from a simple statistical model. This approach was intended to emphasize outbreaks of the disease and mask normal patterns such as seasonal variations and different reporting practices across the states.

In this video we considered the later period of the disease (1980-1988) when the incidence rates have stabilized after the steep drop that was caused by the introduction of vaccination programs at the end of 1960's.

Let z_{ij} be the logarithm of the reported incidence rates in state i for month j (We added 1 before taking logarithm to avoid problems with zero incidence rates). We used median polish to fit state effects $s_i$and time effects t_j. The residuals

$$\eta_{ij} = z_{ij} - s_i - t_j$$

Figure 5.12: Smoothed incidence rates in December 1986

for any particular state looked like a stationary time series except for one or two peaks caused by larger outbreaks.

To emphasize the outbreaks we smoothed out the "small" noise leaving only extreme peaks. We defined an η_{ij} to be unusual if it was in the upper 0.95 quantile of the residuals. We then applied running medians of size 3 in time for every state to the residuals that were not considered unusual, i.e. we chose

$$\hat{\eta}_{ij} = \begin{cases} \text{Median}(\eta_{ij} : |j-k| \leq 3) \text{ if } \eta_{ij} \text{ was not unusual} \\ \eta_{ij} \text{ otherwise.} \end{cases}$$

The resulting animation identifies what one could define as an outbreak of the disease without confusing the scene with the seasonal and between state effects.

5.7 Discussion

Mumps in the US is a seasonal disease. The peak occurs in early spring, while the lowest incidence rates can be observed in autumn. As most of the cases are school age children, this can be in part explained by the school year. Over a longer period the mumps disease had a high incidence rate before the vaccination programs started around 1970. By 1980 these vaccination programs almost completely eradicated the disease, leaving only a few cases per state per month. Some states ceased mandatory vaccination programs at about that time and strong outbreaks of the disease occurred in 1986-1987, primarily among unvaccinated adolescents and young adults in those states. These statements are clearly supported by the graphs in Figure 5.1 of the logarithm of the incidence rates in California and Wisconsin. We can see seasonal periodicity (high in spring and low in autumn) and an outbreak in Wisconsin in the second half of eighties.

Annual periodicity in the incidence rate for mumps can be observed in both the raw data videos and the smoothed versions. The periodic effect is particularly striking in the early years of the data set, before the widespread use of the mumps vaccine reduced the typical monthly incidence rate below .1 (cases per 100,000 people). However, the effect can be discerned throughout the data set, especially in the smoothed version.

The geographic spread of mumps cannot be easily discerned in the raw data; however, repeated viewing eventually allows one to make such an interpretation. The effect is probably most noticeable in the winter of 1987-1988 in the states surrounding Illinois. In the smoothed data the geographic spread of the disease is readily apparent. This is particularly clearly visible during the late winter of 1986-87 when the disease spreads from Illinois to Arkansas and Tennessee and in the subsequent winter when the disease spreads to all the neighboring states.

5.8 Summary

In this chapter animation of mumps incidence rates in the United States was considered. First we described the data which represents counts of the disease for each state for every month from January, 1968 until December, 1988. The data indicate a strong seasonal trend that could be explained by a school year and a decreasing trend due to introduction of the mumps vaccine at the beginning of considered period.

In Section 5.4 the techniques used to animate space time data were discussed. The main features were the mapping of numeric values into color space and the importance of smoothness of the animation in spatial and temporal domains. Some techniques to reduce the amount of computations were also discussed.

In Section 5.5 the results from Chapters 2, 3, and 4 were applied to mumps data. First we considered how the kernel smoothing method described in Chapter 4 can be used to obtain a smooth function of mumps incidence rates over the territory of the United States. Then we estimated the dependence structure of the incidence rates process using methods from Chapters 2 and 3.

In Section 5.6 we describe successively smoother animations of the mumps incidence rates recorded on a videotape. The last animation illustrates a method to detect outbreaks of a disease.

Chapter 6

Contributions and Future Work

The contributions can be shortly summarized by the following list:

1. Estimating dependencies of a spatial process given its integrals.
2. Prediction given integrals.
3. Animation of disease maps.

The real-life data can be often modeled as averages over various regions of some space-time process. The nature of such process is often stochastic and to perform a reasonable statistical analysis we need to get empirical evidence about process dependence structure. The estimation of the the dependence structure of spatial process from aggregate data was not considered before. In Chapter 2 we proposed to estimate an isotropic covariance function of spatial process from integrals of that process.

Having dependence structure of the discussed process we can proceed with various types of prediction. Prediction from aggregate data is somewhat different from prediction using point observations. We discuss such specifics and present a new simple to use procedure to predict from aggregate data in Chapter 4. This procedure is computationally similar to kernel smoothing methods.

We applied the developed theory and algorithms to animate the spread of mumps in the continental United States. The developed methodology can be used to animated other (similarly widespread) diseases. We now have available data for other 19 notifiable diseases for the period from January 1962 until December 1992. Unfortunately, only three of those diseases are relatively widespread, other are extremely rare.

Directions for the future work are

1. Generalizations of the covariance function estimation:

   (a) develop an effective procedure to estimate the covariance function when the data represents integrals over regions of general form in higher than two dimensions.

   (b) develop a method to estimate a generalized covariance functions (not only variogram and covariogram).

   (c) Prove asymptotic properties, e.g., almost sure consistency.

2. Improve prediction theory:

   (a) investigate theoretic properties of the kernel type estimators.

   (b) finding the optimal kernels corresponding to the estimated covariance function.

   (c) obtain uncertainty in prediction due to uncertainty in the estimate of the covariance function.

3. Apply the developed techniques on other datasets.

4. Develop Bayesian and likelihood approach in estimating dependencies and obtaining predictive distributions.

Bibliography

[1] Anderson, T.W., (1958). *An Introduction to Multivariate Statistical Analysis*. John Willey & Sons.

[2] Asimov, D. (1985). The Grand Tour for Viewing Multidimensional Data. *SIAM J. Sci. Stat. Comput.* **6**, 128-143.

[3] Becker, R. A. and Wilks, A. R. (1991). Maps in S. *AT&T Bell Laboratories Statistics Research Report*

[4] Bonnesen, T. and Frenchel, W. (1987). *Theory of Convex Bodies*. BSC Associates.

[5] Busemann, H. (1958). *Convex Surfaces*. Interscience Publishers, London.

[6] Cleveland, W. S. and McGill, M. E. (1988). *Dynamic Graphics for Statistics*. Wadsworth, Inc., Belmont, CA.

[7] Cliff, A.D. and Haggett, P. (1992). *Atlas of Disease Distributions*. Blackwell Publishers.

[8] Cramer, H. and Leadbetter, M.R. (1967). *Stationary and Related Stochastic Processes*. J.Wiley.

[9] Cressie, N. (1985). Fitting Variogramm Models by Weighted Least Squares. *Mathematical Geology*. **17**: 563:568.

[10] Cressie, N. (1991). *Statistics for Spatial Data*. J.Wiley.

[11] Deetz and Adams (1921). Elements of Map Projection. *USGS Special Publication* No. 68, GPO.

[12] Delfner, P. (1976). Linear Estimation of non-Stationary Spatial Phenomena. *Advanced Geostatistics in the Mining Idustry.* 49-68. Reidel. Dordrecht.

[13] Dennis, J.E., Jr., and Schnabel, R.B. (1983). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentince-Hall.

[14] Eddy, W.F. and Mockus, A. (1993a). Noninteractive Display of Functions with Temporal and Spatial Variation (or Surreal-time Graphics). *New Directions in Statistical Data Analysis and Robustness*. 47-60. Birkhäuser Verlag, Basel.

[15] Eddy, W.F. and Mockus, A. (1993b). An Example of Noninteractive Dynamic Graphics for Manufacturing Process Data. *International Statistical Review*. **61**(1): 81-95.

[16] Eddy, W.F. and Mockus, A. (To appear 1994). An Example of the Estimation and Display of a Smoothly Varying Function of Time and Space - The Incidence of the Disease Mumps. *Journal of the American Society for Information Science*. Special Issue on Spatial and Geographic Information Systems.

[17] Gelfand, I.M. and Vilenkin, N. Ya. (1964). *Generalized Functions, Vol. 4, Applications of harmonic analysis*. Academic Press, New York.

[18] Ghosh, M. and Rao, J.N.K. (1994). Small Area Estimation: an Appraisal. *Statistical Science*. **9**: 55-93.

[19] Grünbaum, B. (1967). *Convex Polytopes*. London, Interscience.

[20] Handcock, M.S. and Wallis J.R. (1994). An Approach to Statistical Spatial-Temporal Modeling of Meteorological Fields. *JASA*. **89**: 368-390.

[21] IMSL Version 2.0 (1991). *Users Manual, Math/Library*.

[22] Jones, R.H. and Vecchia, A.V. (1993). Fitting Continous ARMA Models to Unequally Spaced Spatial Data. *JASA*. **88**: 947-954.

[23] Kitanidis, P.K. (1983). Statistical Estimation of Polynomial Generalized Covariance Function in Hydrologic Applications. *Water Resources Research*. **19**:909-921.

[24] Kitanidis, P.K. (1986). Parameter Uncertainty in Estimation of Spatial Functions: Bayesian Analysis. *Water Resources Research*. **22**:499-507.

[25] Laslett, G.M. (1994). Kriging and Splines: an Empirical Comparison of Their Predictive Performance in Some Applications. *JASA*. **89**: 391-409

[26] Levkowitz, H. and Herman, G.T. (1992). Color Scales in Image Data. *IEEE Computer Graphics & Applications*. January, 1992. 72-80.

[27] Marshall, R.J. and Mardia, K.V. (1985). Minimum Norm Qadratic Estimation of Components of Spatial Covariances. *Mathematical Geology*. **17**:517-525.

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

[29] Matheron, G. (1973). The Intrinsic Fandom Functions and Their Applications. *Adv. Appl. Prob.*. **5**:439-468.

[30] Mockus, J. B. (1989). *Bayesian Approach to Global Optimization*. Kluwer Academic Publishers, Doderecht, Holland 1989.

[31] O'Sullivan, F. (1986). A Statistical Perspective on Ill-posed Problems. *Statistical Science*. **1**: 502-527.

[32] Pavlidis, T. (1982). *Algorithms for Graphics and Image Processing.* Computer Science Press, Rockville, MD.

[33] Santalo, L.A. (1976). *Integral Geometry and Geometric Probability* in Encyclopedia of Mathematics Vol. 1. Addison Wesley.

[34] Schittkowski, K. (1986). NLPQL: A FORTRAN Subroutine Solving Constrained Nonlinear Programming Problems. (edited by Clyde L. Monma). *Annals of Operations Research*. **5**:485-500.

[35] Stein, M.L. (1990). Uniform Asymptotic Optimality of Linear Predictions of a Random Field Using an Incorrect Second-Order Structure. *Ann. Statist.*. **18**: 850-872.

[36] Stein, M.L. (1991). A Kernel Approximation to the Kriging Predictor of a Spatial Process. *Ann. Inst. Statist. Math.* **43**:61-75.

[37] Tobler, W.R. (1979). Smooth Pycnophylactic Interpolation for Geographic Regions. *Journl of the American Statistical Association*. **74** 367: 519-536.

[38] Tobler, W.R., Kennedy, S. (1985). Smooth Multidimensional Interpolation. *Geographical Analysis.* **17**(3): 251-257.

[39] Tufte, E.R. (1983). *The Visual Display of Quantitative Information*. Graphics Press. Cheshire, Connecticut.

[40] Whaba, G. (1990). *Spline Models for Observational Data*. SIAM.

[41] Whittle, P. (1954). On Stationary Processes in the Plane. *Biometrika*. **41**: 434-449.

[42] Vecchia (1985). A General Class of Models for Stationary Two-Dimensional Random Processes. *Biometrika*. **72**: 281-291.

[43] Zimmerman, D.L. and Cressie, N. (1992). Mean Squared Prediction Error in the Spatial Linear Model with Estimated Covariance Parameters, *Ann. Inst. Statist. Math.,* **44**:27-43.

Appendix A

Appendix

A.1 Integral Equations for Variogram

Let $f(x)$ be a zero mean stochastic process on R^d. If a quantity $\rho(u, v) = E((f(u) - f(v))^2)$ is finite and is only a function of $u - v$, i.e $\rho(u, v) = \rho(u - v)$ then we say that f has variogram ρ.

The process having a variogram can be non-stationary, while for a stationary process the variogram always exists.

Using the notation from Section 2.2 we will express the expected value of $(z_i - z_j)^2$ in terms of the variogram.

$$
\begin{aligned}
&E((z_i - z_j)^2) = \\
&\quad E\left[\left(\int_{A_i}\int_{A_j} f(u) - f(v) du dv\right)^2\right] \\
&\quad = E\left[\int_{A_i,A_j,A_i,A_j} (f(u_1) - f(v_1)(f(u_2) - f(v_2) du_1 \dots dv_2\right] \\
&\quad = E\left[\int_{A_i,A_j,A_i,A_j} \sum_{k,l=1,2} (-1)^{k-l} f(x_{k,1}) f(x_{l,2}) du_1 \dots dv_2\right] \\
&\quad = -E\left[\frac{1}{2}\int \sum_{k,l=1,2} (-1)^{k-l} \left((f(x_{k,1}) - f(x_{l,2}))\right)^2 du_1 \dots dv_2\right] \\
&\quad +E\left[\frac{1}{2}\int \sum_{k,l=1,2} (-1)^{k-l} \left(f(x_{k,1})^2 + f(x_{l,2})^2\right) du_1 \dots dv_2\right]
\end{aligned}
$$

$$= -\frac{1}{2} \int_{A_i, A_j, A_i, A_j} \sum_{k,l=1,2} (-1)^{k-l} \rho(x_{k,1} - x_{l,2}) du_1 \dots dv_2 \tag{A.1}$$

Now we can use sample values $(z_i - z_j)^2$ to solve Equation (A.1) for ρ.

A.2 Kinematic Measure

Here I will follow Santalo 1976 pp. 80–92 and Bonnensen and Frenchel 1987 pp. 74-75. First lets consider a simple example.

Example 2: Let K be any ordered pair of points $(\mathbf{a}, \mathbf{b})$ in R^2 with a fixed distance apart $\|\mathbf{a} - \mathbf{b}\| = l$ (an oriented stick). The movements of the stick can be parametrized in terms of the displacement $\mathbf{q}$ of its first end $\mathbf{a}$ and by rotation α with respect to its original orientation. The new Cartesian coordinates for the ends of the stick after the movement $(\mathbf{q}, \alpha)$ are $(\mathbf{a} + \mathbf{q}, \mathbf{a} + \mathbf{q} + (l\cos(\alpha), l\sin(\alpha)))$ assuming that the vector $\mathbf{b} - \mathbf{a}$ was oriented in the direction of x coordinate axis.

We might be interested in some subset of all possible movements of the stick (all possible movements are $\mathbf{q} \in R^2,\ \alpha \in [0, 2\pi)$). For example, we consider all movements of the stick K such that it has first end inside a circle with radius r and center at the origin. Using the parametrizations of the movements we need to find all pairs $\mathbf{q}, \alpha$ so that $\|\mathbf{a} + \mathbf{q}\| < r$ and $\alpha \in [0, 2\pi)$. Integrating all those movements with respect to $d\mathbf{q}d\alpha$ we get the "measure" of all such movements to be $2\pi r^2 \times 2\pi$

Let a set M of geometric objects be given. For example, a set of points, a set of lines in plane or in space, a set of planes, a set of point pairs with a fixed distance, etc. To such a set we assign a "measure". Let an element of M (point, line, plane) be determined by independent coordinates $(\alpha_1, \dots, \alpha_k)$. Let $f(\alpha_1, \dots, \alpha_k)$ or briefly $f(\alpha)$ be an (initially arbitrary) positive function of α. By the measure $\mu(M)$ of M we mean the integral $\int f(\alpha)d\alpha$ extended over M (M and f are supposed to be such that this integral is meaningful). The usual condition imposed on the arbitrary "density function" f is that the measure of a set M remains unchanged under motions. In other words: If the set M goes to $\overline{M}$ by means of a motion, then we should have $\mu(M) = \mu(\overline{M})$. This requirement identifies function f up to an arbitrary positive factor, for sets of points, lines, and planes.

We will consider points and sets of points on the Euclidean plane with rectangular system of Cartesian coordinates. A motion is defined as a transformation of the plane onto itself $u : P(x, y) \rightarrow$

$P'(x', y')$ represented by the equations

$$x' = x\cos\phi - y\sin\phi + a$$
$$y' = x\sin\phi + y\cos\phi + b$$

where a, b, ϕ are parameters that have the following respective ranges:

$$-\infty < a < \infty, \ \ -\infty < b < \infty, \ \ 0 \le \phi \le 2\pi.$$

Let a pair of points A, B with a fixed distance l apart (we will refer to it as "a segment") be given. Let x, y be Cartesian coordinates of the point A and α be the angle between the ray $\overrightarrow{AB}$ and the $\vec{x}$ axis of the coordinate system. Three components (x, y, α) will fully specify the position of the oriented segment AB.

A kinematic density function f (of the segment AB) invariant under the motion u would satisfy the equation

$$\int_M f(x, y, \alpha)dxdyd\alpha = \int_{M'} f(x, y, \alpha)dxdyd\alpha \tag{A.2}$$

where M' is obtained from M by means of the motion u

$$x' = x\cos\phi - y\sin\phi + a$$
$$y' = x\sin\phi + y\cos\phi + b$$
$$\alpha' = (2\pi + \alpha - \phi) \bmod 2\pi$$

Arbitrariness of the movement u and Equation (A.2) imply that $f(x, y, \alpha)$ is a constant. So the kinematic density for the segment is a constant.

To determine that constant and obtain relation to the weight function $W_{A_iA_j}(l)$ consider transformation of variables in Equation (2.4). The transformation is from Cartesian coordinates (x_1, y_1, x_2, y_2) to the coordinates (x_1, y_1, l, ϕ), where $l = ((x_1-x_2)^2+(y_1-y_2)^2)^{\frac{1}{2}}$, $\phi = \tan((y_1-y_2)/(x_1-x_2))$. The Jacobian is

$$\begin{vmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & \cos(\phi) & -l\sin(\phi) \\ 0 & 0 & \sin(\phi) & l\cos(\phi) \end{vmatrix} = l \tag{A.3}$$

Let $A \cap B = \emptyset$ and let $M(l, A, B)$ be the measure of all movements of an oriented line segment of length l with one end in the set A and the other in the set B. Let $u = (x_1, y_1), \ v = (x_2, y_2)$ then using Equation (A.3) we get

$$\begin{aligned}
W_{AA}(l) &= \int_{u\in A, v\in A, \|u-v\|=l} dudv \\
&= \int_{(x_1,y_1)\in A,(x_1+l\cos(\phi),y_1+l\sin(\phi))\in A} l dx_1 dy_1 d\phi \\
&= M(l,A,A)l, \qquad \text{(A.4)} \\
W_{AB}(l) &= \frac{1}{2}\left(\int_{u\in A, v\in B, \|u-v\|=l} dudv + \int_{u\in B, v\in A, \|u-v\|=l} dudv\right) \\
&= \frac{1}{2}\int_{(x_1,y_1)\in A,(x_1+l\cos(\phi),y_1+l\sin(\phi))\in B} l dx_1 dy_1 d\phi \\
&\quad +\frac{1}{2}\int_{(x_1,y_1)\in B,(x_1+l\cos(\phi),y_1+l\sin(\phi))\in A} l dx_1 dy_1 d\phi \\
&= M(l,A,B)l/2, \ \ A\cap B = \emptyset \qquad \text{(A.5)}
\end{aligned}$$

the two cases are different because obtaining W (Equation (1.1)) we specify which end of the segment is in which set.

A.3 A Segment Intersecting Two Other Segments

Equation (3.5) is obtained in Santalo 1976. Equations (3.6) and (3.7) are obtained similarly, so I will derive only equation (3.6). For Equation (3.6) I will consider the situation as shown in Figure A.1. I will first fix angle ϕ or $\pi-\phi$ the segment K has with ray $R(A\infty)$ and calculate the measure of all non-rotational movements $M(\phi)$ (for two fixed orientation of K) so that the segment crosses rays $R(A\infty)$ and $R(OG)$. Then I will integrate the $M(\phi)+M(\pi-\phi)=2M(\phi)$ over the range of possible values for ϕ when $\phi\subset[0,\pi]$.

From Figure A.1, $M(\phi)$ is equal to the area of the triangle $\widehat{EFG}$. So $2M(\phi) = (|AF| - |AE|)|AH|\sin(\phi)$. In the notation of Equation (3.6):

$$\begin{aligned}
|OA| &= l_2 \\
|AF| &= |HG| = l \\
|OH|\sin(\alpha) &= |HG|\sin(\phi-\alpha) \\
|OA|/|OH| &= |AE|/|HG|
\end{aligned}$$

Using those equations we get

$$\begin{aligned}
(|AF|-|AE|)|AH| &= (|AF|-|AE|)(|OH|-|OA|) \\
&= ll_2(1-C)(1/C-1),
\end{aligned}$$

Figure A.1: The area of the filled triangle $\widehat{EFG}$ is the measure of all non-rotational movements of the segment K when it intersects $R(A\infty)$ and $R(OG)$

where

$$\begin{aligned} C &= |OA|/|OH| = l_2/|OH| \\ &= l_2/\left(|HG|\sin(\phi-\alpha)/\sin(\alpha)\right) \\ &= \frac{l_2\sin(\alpha)}{l\sin(\phi-\alpha)} \end{aligned}$$

and hence

$$2M(\phi) = \sin(\phi)\left(l - \frac{l_2\sin(\alpha)}{\sin(\phi-\alpha)}\right)\left(\frac{l\sin(\phi-\alpha)}{\sin(\alpha)} - l_2\right), \tag{A.6}$$

and integrand in Equation (3.6) follows.

To get the limits of integration in Equation (3.6) consider extreme orientations of the segment K when it can intersect $R(A\infty)$ and $R(OG)$. Minimal angle ϕ_0 can be obtained using equation $l_2\sin(\alpha) = l\sin(\phi_0-\alpha)$, and maximal angle ϕ_1 is either π (if $l_2 < l$) or $\pi - \phi_0$.

A.4 Consistency of the Step Estimators

Let $f(x)$ be a stationary zero-mean stochastic process on R^2 with an isotropic covariance function γ (i.e., $\gamma(x_1, x_2) = \gamma(\|x_1 - x_2\|)$ is a function of the distance between x_1 and x_2). The observations

z_i of this process have following form: $z_i = \int_{A_i} f(x)dx$, where $A_i, i = 1, \ldots, N$ partition region A.

Let $l_k,\ k = 0, \ldots, K$ be an ordered sequence of real numbers with $l_0 = 0$. A piecewise constant estimator $\hat{\gamma}(l) = \sum_k \gamma_k I_{[l_k, l_{k+1})}(l)$ of the covariance function is given by following equation:

$$\hat{\gamma} = \arg\min_{\gamma} \sum_{i,j} \left(C_{ij}(z_i z_j - \int \gamma(l) W_{ij}(l)dl) \right)^2 \tag{A.7}$$

where C_{ij} are weights (for example, $C_{ij} = \frac{1}{S_{A_i} S_{A_j}}$ where S_{A_i} is the area of the region A_i) and W is defined by Equation $\int_A \int_B dudv = \int_0^\infty W_{AB}(l)dl$ for any two sets A and B. Let $p_{ij,k} = C_{ij} \int_{l_k}^{l_{k+1}} W_{ij}(l)dl$ and $Z_{ij} = C_{ij} z_i z_j$. Then Equation (A.7) can be rewritten as

$$\arg\min_{\gamma_k,\, k=1,\ldots,K} \sum_{i,j} \left(Z_{ij} - \sum_l \gamma_l p_{ij,l} \right)^2 \tag{A.8}$$

The solution to this quadratic minimization problem is the same as the solution to the system of linear equations (we assume the matrix $P^T P$ to be invertible)

$$P^T P \boldsymbol{\gamma} = P^T Z, \tag{A.9}$$

where the vector $\boldsymbol{\gamma} = (\gamma_l)$, the matrix $P = (p_{ij,l})$ (note, that ij is an index for rows and l is an index for columns), and the vector $Z = (Z_{ij})$.

We will consider asymptotic behavior of a sequence of estimators $\boldsymbol{\gamma}^{(q)}$ as $q \to \infty$. To simplify the notation we will omit the index (q) in this sequence. All asymptotic conditions will be given in terms of this implicit index (q).

The unique solution to Equation (A.9) is $(P^T P)^{-1} P^T Z$ if $P^T P$ is invertible. Let $M = (P^T P)^{-1} P^T$. The expected value of

$$\begin{aligned}
E(\boldsymbol{\gamma}) &= \left(P^T P\right)^{-1} P^T E(Z) \\
&= M \int \left(C_{ij} W_{ij}(l)\right) \gamma(l)\, dl \\
&= M \sum_k \int_{l_k}^{l^{k+1}} \left(C_{ij} W_{ij}(l)\right) \gamma(l)\, dl \\
&= M \sum_k \int_{l_k}^{l_{k+1}} \left(C_{ij} W_{ij}(l)\right) (\gamma(l_k) + \xi_k(l))\, dl \\
&= M \sum_k (p_{ij,k})\, \gamma(l_k) + M \sum_k \int_{l_k}^{l_{k+1}} \left(C_{ij} W_{ij}(l)\right) \xi_k(l)\, dl \\
&= \left(P^T P\right)^{-1} P^T P(\gamma(l_k)) + M \sum_k \int_{l_k}^{l_{k+1}} \left(C_{ij} W_{ij}(l)\right) \xi_k(l)\, dl
\end{aligned} \tag{A.10}$$

where $\xi_k(l) = \gamma(l_k) - \gamma(l)$. Let $\xi_k = \sup_{l_k<l<l_{k+1}} |\xi_k(l)|$ then if $\max_k \xi_k < \infty$

$$|E(\gamma_k) - \gamma(l_k)| < \xi_k \tag{A.11}$$

In particular, when $\gamma(l)$ satisfies a Lipschitz condition (i.e., there exist a constant $h < \infty$ such that for $\forall\epsilon > 0\ \exists\delta > 0\ \ : \forall l$ and $\forall x < \delta,\ |\gamma(l) - \gamma(l+x)| < h\epsilon$), $P^T P$ is invertible, and $\max_k(l_{k+1} - l_k) \to 0$ the estimator is asymptotically unbiased as stated in the following Proposition.

Proposition A.4.1 *Let $\gamma(l)$ satisfy a Lipschitz condition, $(P^{(q)})^T P^{(q)}$ be a sequence of invertible matrices, $\lim_{(q)} \max_k(l_{k+1}^{(q)} - l_k^{(q)}) = 0$, $l_K^{(q)} \geq l_K$, and $l_0^{(q)} \leq l_0$. Then $\lim E(\hat{\gamma}^{(q)}(l)) = \gamma(l)$ for any $l_0 < l < l_K$*

The covariance matrix for the estimator $\boldsymbol{\gamma}$ is

$$\begin{aligned}
E\left(\boldsymbol{\gamma}\boldsymbol{\gamma}^T\right) - E(\boldsymbol{\gamma})\,E\left(\boldsymbol{\gamma}^T\right) &= E\left(MZZ^TM^T\right) - ME(Z)\,E\left(Z^T\right)M^T \\
&= E\left(MPP^-ZZ^T\left(P^T\right)^- P^TM^T\right) - MPP^-E(Z)\,E\left(Z^T\right)\left(P^T\right)^- P^TM^T \\
&= E\left(P^-ZZ^T\left(P^T\right)^-\right) - P^-E(Z)\,E\left(Z^T\right)\left(P^T\right)^- \\
&= \left(\sum_{(ij),(mn)} p^-_{ij,k}p^-_{mn,l}E\left(Z_{ij}Z_{mn}\right)\right) - \left(\sum_{(ij),(mn)} p^-_{ij,k}p^-_{mn,l}E\left(Z_{ij}\right)E\left(Z_{mn}\right)\right) \\
&= \left(\sum_{(ij),(mn)} p^-_{ij,k}p^-_{mn,l}\left(E\left(Z_{im}\right)E\left(Z_{jn}\right) + E\left(Z_{in}\right)E\left(Z_{jm}\right)\right)\right)
\end{aligned}$$

where $P^- = \left(p^-_{ij,k}\right)$ is Moore-Penrose inverse of the matrix P. The last equality was obtained using a formula for the fourth centered moment of the multivariate normal distribution (see, e.g., Anderson pp. 39).

The quantity

$$\sum_{(ij),(mn)} p^-_{ij,k_1}p^-_{mn,k_2}E(Z_{im})E(Z_{jn}) = \sum_{(ij),(mn)} p^-_{ij,k_1}p^-_{mn,k_2}$$
$$\left(\sum_k p_{im,k}\gamma(l_k) + \int_{l_k}^{l_{k+1}} C_{im}W_{im}(l)\xi_k(l)dl\right)\left(\sum_k p_{jn,k}\gamma(l_k) + \int_{l_k}^{l_{k+1}} C_{jn}W_{jn}(l)\xi_k(l)dl\right) \tag{A.12}$$

is the first term of the the covariance between γ_{k_1} and γ_{k_2} (the second term has the same expression only the subscript im is changed to in and the subscript jn is changed to jm. If $\gamma(l)$ satisfies a Lipschitz condition and $\max_k(l_{k+1} - l_k) = 0$, the expression A.12 can be simplified as the term including $\xi_k(l)$ is much smaller than the term including $\gamma(l)$. The result is:

$$\sum_{(ij),(mn)} p^-_{ij,k_1}p^-_{mn,k_2}E(Z_{im})E(Z_{jn}) \approx \sum_{(ij),(mn)} p^-_{ij,k_1}p^-_{mn,k_2}\left(\sum_u p_{im,u}\gamma(l_u)\right)\left(\sum_v p_{jn,v}\gamma(l_v)\right) \tag{A.13}$$

Proposition A.4.2 *Let $\gamma(l)$ satisfy a Lipschitz condition, $(P^{(q)})^T P^{(q)}$ be a sequence of invertible matrices, $\lim_{(q)} \max_k (l^{(q)}_{k+1} - l^{(q)}_k) = 0$, $l^{(q)}_K \geq l_K$, $l^{(q)}_0 \leq l_0$, and the sum*

$$\sum_{(ij),(mn),u,v} p^{(q)-}_{ij,k_1} p^{(q)-}_{mn,k_2} \left(p^{(q)}_{im,u} p^{(q)}_{jn,v} + p^{(q)}_{in,u} p^{(q)}_{jm,v} \right) \gamma(l^{(q)}_u) \gamma(l^{(q)}_v)$$

tends to zero as $q \to \infty$.

Then the covariance of the estimator $\lim_{q\to\infty} Cov(\hat{\gamma}^{(q)}(l_1), \hat{\gamma}^{(q)}(l_2)) = 0$ *for any* $l_0 < l_1, l_2 < l_K$.

To investigate the last condition in the proposition we need to discuss properties of the matrix P which depend on the geometry of regions A_i. We will use two measures for the size of a region in R^n.

Definition A.4.3 *The outer diameter (or diameter) of a region A is*

$$D(A) = \sup_{x\in A, y\in A} \|x - y\|$$

where $\|\cdot\|$ is Euclidean distance. In words: the outer diameter of A is the diameter of the smallest sphere containing A.

The inner diameter of a region A is

$$d(A) = \sup_{c(R)\subset A} R$$

where $c(R)$ is a sphere of diameter R in R^n. In words: the inner diameter of A is the diameter of the largest sphere contained in A.

The inner distance between A and B is

$$d(A, B) = \inf_{x\in A, y\in B} \|x - y\|.$$

The outer distance between A and B is

$$D(A, B) = \sup_{x\in A, y\in B} \|x - y\|.$$

Let $D = \max_i D(A_i)$, and $d = \min_i d(A_i)$.

We will assume several conditions on the asymptotic behavior of the regions and on the smoothness of the covariance function. Those assumptions can be relaxed.

1) γ satisfies a Lipschitz condition. It is a sufficient condition for the sample paths to be differentiable (see, e.g., Cramer and Leadbetter, pp. 125).

2) $l_K/d(A) \to 0$.

3) $D \to 0$.

4) $D/d = O(1)$.

5) $\max_k(l_k - l_{k-1}) \to 0$.

6) $\frac{D}{\min_k(l_k - l_{k-1})} \to 0$

Condition (1) and (5) together with Equation (A.11) imply asymptotic unbiasedness.

To prove consistency we need to show that the variance of γ_k goes to zero.

Condition (6) makes the matrix $P^T P$ asymptotically tri-diagonal as $p_{ij,k}p_{ij,l} \neq 0$ only when $|k-l| \leq 1$. The non-diagonal elements are small, so $\gamma_k \approx \frac{\sum_{ij} p_{ij,k} Z_{ij}}{\sum_{ij}\left(p_{ij,k}\right)^2}$ and

$$\begin{aligned} E\left(\gamma_k^2\right) - E(\gamma_k)^2 &= \frac{1}{\left(\sum_{ij} p_{ij,k}\right)^2} \sum_{ij,mn} p_{ij,k} p_{mn,k} E\left(Z_{ij}Z_{mn}\right) - E(Z_{ij})E(Z_{mn}) \\ &\approx \quad 1/C^2 \sum_{ij,mn,v,u} p_{ij,k}p_{mn,k}\left(p_{im,u}p_{jn,v} + p_{in,u}p_{jm,v}\right)\gamma(l_u)\gamma(l_v) \end{aligned}$$

where $C = \frac{1}{\left(\sum_{ij}\left(p_{ij,k}\right)^2\right)}$. Compare this expression with Equation A.13. Now we will show that

$$1/C^2 \sum_{ij,mn,v,u} p_{ij,k}p_{mn,k}p_{im,u}p_{jn,v}\gamma(l_u)\gamma(l_v) \tag{A.14}$$

tends to zero under the conditions (1)-(6). The basic idea is that the matrix $p_{im,u}$ contains mostly zero entries. As the covariance function is bounded (it is automatically bounded when it satisfies a Lipschitz condition) it is enough to show that the sum given by $1/C^2 \sum_{ij,mn,v,u} p_{ij,k}p_{mn,k}p_{im,u}p_{jn,v} \to 0$. Condition (6) implies that the matrix $p_{ij,k}$ has no more than three nonzero entries in one row (the rows are indexed by ij), Condition (2) implies that most of the rows are all zeros as only the rows corresponding to the pairs of regions that have $D(A_i, A_j) < l_K$ can have nonzero entries. If the total number of regions is n, then the bounds on the number of pairs of regions satisfying this condition are from $n \times (\pi l_K^2/D^2)$ to $n \times (l_K^2/d^2)$, i.e., for each region A_i we draw a circle with radius l_K and the smallest region has diameter d so there are no more than $\pi l_K^2/\frac{\pi d^2}{4}$ regions inside this circle. The biggest region has diameter D so there are no less than $(\pi l_K^2/D^2)$ regions inside the circle.

When we take the product $\sum_{ij,mn,v,u} p_{ij,k} p_{mn,k} p_{im,u} p_{jn,v}$ we can just sum only over the set of indexes ij and mn such that $D(A_i, A_j) < l_K$ and $D(A_m, A_n) < l_K$. In addition, we are can restrict the sum to terms when $\sum_u p_{im,u} \neq 0$ and $\sum_v p_{jn,v} \neq 0$. Those we have four conditions for a term indexed by (ij, mn) to be nonzero: $D(A_i, A_j) < l_K$, $D(A_m, A_n) < l_K$, $D(A_i, A_m) < l_K$, $D(A_n, A_m) < l_K$. We can calculate that there are no more than $n \times 4l_K^2/d^2 \times 4l_K^2/d^2 \times 4l_K^2/d^2 = n\left(\frac{4l_K}{d}\right)^6$ terms that are nonzero, i.e., we can take n different regions A_i and for each region A_i there are no more than $4l_K^2/d^2$ regions A_j distance l_K or less away , and no more than $4l_K^2/d^2$ regions A_m distance l_K or less away. For each triplet A_i, A_j, A_m there are no more than $4l_K^2/d^2$ regions A_n distance l_K or less away from both A_m and A_j. The sum in the divisor $C^2 = \left(\sum_{ij} (p_{ij,k})^2\right)^2 = \sum_{ij,mn} (p_{ij,k})^2 (p_{mn,k})^2$ has no less than $(n \times (\pi l_K^2/D^2))^2 = \pi^2 n^2 \left(\frac{l_K}{D}\right)^4$ nonzero terms. The ratio of the number of nonzero terms in the numerator and the denominator of Equation (A.14) tends to zero

$$\begin{aligned} \frac{n\left(\frac{4l_K}{d}\right)^6}{\pi^2 n^2 \left(\frac{l_K}{D}\right)^4} &= \frac{4^4}{\pi^2}\frac{4l_K^2/d^2}{n}\frac{D^4}{d^4} \\ &< \frac{4^6}{\pi^2}\frac{l_K^2/d^2}{d(A)^2/d^2}\frac{D^4}{d^4} \\ &= \frac{4^6}{\pi^2}\frac{l_K^2}{d(A)^2}\frac{D^4}{d^4} \to 0. \end{aligned} \tag{A.15}$$

using Condition (2). To conclude the proof that Equation (A.14) tends to zero we notice that the number of terms in the C that are not of the order of 1 (for $C_{ij} = 1/(S_{A_i} S_{A_j})$) is negligible with respect to the total number of terms. The term $p_{ij,k} = C_{ij} \int_{l_k}^{l_{k+1}} W_{A_i A_j}(l)dl$. The quantity $\int_{l_k}^{l_{k+1}} W_{A_i A_j}(l)dl = S_{A_i} S_{A_j}$ if $D(A_i, A_j) \leq l_{k+1}$ and $d(A_i, A_j) \geq l_k$. For such pairs of regions taking $C_{ij} = 1/(S_{A_i} S_{A_j})$ the $p_{ij,k} = 1$. When $0 < p_{ij,k} < 1$ we must have asymptotically $D(A_i, A_j) \geq l_{k+1}$ and $d(A_i, A_j) \leq l_{k+1}$ or $D(A_i, A_j) \geq l_k$ and $d(A_i, A_j) \leq l_k$ as $D/\max_k(l_{k+1} - l_k) \to 0$ by condition (5). The number of pairs of regions tat satisfy either relationship is no more than $\sum_k \left((l_k + D)^2 - (l_k - D)^2\right)/d^2 = 4D\sum_k l_k/d^2 < 4DKl_K/d^2$. The total number of $p_{ij,k} > 0$ is no less than $\pi l_K^2/D^2$. So the ratio is no greater than $\frac{4DKl_K/d^2}{\pi l_K^2/D^2} = 4/\pi \frac{D}{l_K/K} D^2/d^2 > 4/\pi \frac{D}{\min_k(l_{k+1} - l_k)} D^2/d^2 \to 0$ by Condition (6).

Appendix B

Appendix

B.1 Software to Calculate the Weight Functions W

The weight function W is important when estimating a covariance function or simulating integrals of a stochastic process when the covariance function is known. The function W is defined as the following integral:

$$W_{AB}(l) = \int_{u \in A, v \in B, \|u-v\|=l} dudv$$

The function $W_{A_iA_j}(t)$ is obtained using Theorem 3.5.4, and Equations (3.9, 3.5, 3.6, 3.7, 3.8). The function $\mathrm{KMeasure}(l, N_A, N_B, \partial A, \partial B)$ (in file geom.cc) calculates a kinematic measure of a line segment of length l with its first endpoint inside the set A and the second endpoint inside the set B. This quantity is exactly $W_{AB}(l)/l$. The function requires that either $A \cap B = \emptyset$ or that $A = B$. The simple generalization of the function for arbitrary two sets A and B can be implemented using Theorem 3.5.4.

The precision (and correctness) of the code can be checked using the relationship:

$$\int_0^\infty W_{AB}(l)dl = S_A S_B$$

where S_A, S_B are areas of the polygons A, B. Those areas can be calculated exactly and compared with the integral on the left. We took two maps of the continental United States (Figures 5.4, and 5.5). Both maps contain 49 regions (there are two separate regions for Michigan state) and we computed all $49(49_1)/2 = 1225$ different integrals $I_{ij} = \int_{a_{ij}}^{b_{ij}} W_{A_iA_j}(l)dl$ and compared them to the exact values $S_{A_i}S_{A_j}$. The boundaries for the integration were taken $a_{ij} = \inf_{u \in A_i,\, v \in A_j} \|u-v\|$ and $b_{ij} = \sup_{u \in A_i,\, v \in A_j} \|u - v\|$. The integrals were computed using Gauss quadrature with Legendre

Percentile	0	25	50	75	100
Figure 5.5	-0.00821244	-2.04368e-05	8.01375e-07	2.33429e-05	0.00302665
Figure 5.4	-0.00263325	-1.27446e-05	-2.73902e-08	1.02901e-05	0.00392718

Table B.1: Percentiles of the relative error

weights (software to compute those quadratures can be obtained from netlib using command `echo "send gaussq from go" | mail netlib@ornl.gov`) using 81 points to evaluate the integral. Several percentiles of the relative error $\frac{I_{ij}-S_{A_i}S_{A_j}}{S_{A_i}S_{A_j}}$ (from a sample of 1225) are in Table B.1

B.2 Software to Estimate a Covariance Function

The software is designed to estimate the covariance of an isotropic zero mean stationary process in R^2 given integrals z_i of the process f over disjoint regions A_i. In practice the "zero mean" is achieved by removing the trend via local or global averages (see Chapter 4).

The calculations are done in two steps. The first step requires borders of the regions A_i and pre-calculates the quantities $W_{ij}^k = W_{A_iA_j}(l_{ij,k})$ for a set of values $l_{ij,k}$ defined by Gauss-Legendre quadrature on intervals $[\inf_{u\in A_i,\, v\in A_j} \|u-v\|, \sup_{u\in A_i,\, v\in A_j} \|u-v\|]$.

In the next step of the program we take the values z_i, $l_{ij,k}$, and W_{ij}^k and solve Equation (2.7).

The minimization is performed as follows:

1. Initial point η_0 is chosen to be used in the next step by the local minimization procedures. The point is chosen by using "bayes1" global minimization routine (see Mockus (1989)).

2. Two different local optimization procedures are then used to improve the estimate. The first procedure uses NLPQL algorithm described in Schittkowski 1986 and the second procedure uses quasi-Newton method (see, e.g., Dennis and Schnabel (1983)) and active set strategy (see, e.g., IMSL (1991)).

3. The best result (the one that minimizes the sum of squares) is chosen.

B.2.1 Forms of the Covariance Function

The various parametric and nonparametric forms of the isotropic covariance function are described in Section 2.2 and Section 2.3.1.

Appendix C

Code Listing

C.1 kmeasure.h

```
#ifndef KMEASURE_H
#define KMEASURE_H

#  define M_PI            3.14159265358979323846

/* A measure of a segment l
 *    intersecting parallel segments l1 and l2
 *    which are distance d apart and shifted by s
 */
double parallelM  (double l, double l1, double l2, double d, double s);

/* A measure of a segment l intersecting two rays
 *    starting at the same point with an angle alpha
 */
double fullM        (double l, double alpha);

/* The same as fullM, but we remove pieces of each ray of length
 *    l1 and l2 correspondingly. The removed pieces start at the origin.
 */
double OutMeasure (double l, double l1, double l2, double alpha);

inline double dmin (double a, double b){ if (a < b) return a; else return b; }
inline double dmax (double a, double b){ if (a > b) return a; else return b; }
```

```
#endif
```

C.2 geom.h

```
#ifndef GEOM_H
#define GEOM_H

typedef struct {
    int x, y;
} intPoint;

#ifdef __cplusplus
extern "C" {
#endif

/* Calculate Kinematic measure of the movements of an oriented segment of
*       length l so that it starts and ends within nonintersecting
*       polygons a and b accordingly.
*       All polygons must have boundary vertices with same (clockwise
* or counter-clockwise) orientation.
*/
double KMeasure         (double l, int an, int bn, intPoint * a, intPoint * b);

/* Area of the region */
double area             (int NPoints, intPoint * p);

/* Perimeter of the region */
double perimeter          (int NPoints, intPoint * p);

/* Is the boundary of the region clockwise? */
int        isClockwise (int NPoints, intPoint * p);

/* Inner distance between the curve a and the curve b */
double innerDistance (int na, int nb, intPoint * a, intPoint * b);

#ifdef __cplusplus
}
#endif
```

```
#ifdef __cplusplus
#include "kmeasure.h"

inline double distance2 (intPoint& a, intPoint& b)
{
   return (a .x − b .x) ∗ (a .x − b .x) +
      (a .y − b .y) ∗ (a .y − b .y);
}

class Point {
public:
   double x, y;
   Point (double xx=0, double yy=0) { x = xx; y = yy;};
   Point (Point& p) { x =  p .x; y = p .y;};
   Point (intPoint& p) { x =  p .x; y = p .y;};
   Point operator− (Point& p)  { Point ans (x − p .x, y − p .y); return ans; };
   Point operator+ (Point& p)  { Point ans (x + p .x, y + p .y); return ans; };
   int   operator== (Point& p) { return (x == p .x && y == p .y);};
   int   operator> (Point& p)  { return (x >= p .x) && (y >= p .y);};
   int   operator< (Point& p)  { return (x <= p .x) && (y <= p .y);};

   double norm () { double tmp = (x∗x + y∗y); return tmp; };
   double operator∗ (Point& p) { return (x ∗ p .x + y∗ p .y); };
   int  between (Point& a, Point& b);
};

class Line{
public:
   double a, b, c;
   Line (Point& a, Point& b);
   Point intersect (Line& line);
   int parallel (Line& line);
   double distance (Point& point);
};


class Segment {
public:
   Point from, to;
   Segment (Point& f, Point& t) : from (f), to (t) {;};
   double innerDistance (Point& p);
   double outerDistance (Point& p);
   double innerDistance (Segment& s);
   double outerDistance (Segment& s);
};


double KMeasureSegment (double l, Segment& a, Segment& b);
int Between (Point& O, Point& a, Point& b);
#endif

#define MIN(a, b) (((a) < (b)) ? (a) : (b))
#define MAX(a, b) (((a) < (b)) ? (b) : (a))

#endif
```

C.3 A.h

```
#ifndef A_H
#define A_H

void quadrature (double x1, double x2, double *x, double *w, int n);

typedef struct {
        int nPoints;
        int minX, minY, maxX, maxY;
        intPoint * line;
} Border;


class Covariance {
public:
   int type; // 0 for sigma e^(alpha x),
             // 1 for step function
   int nBins;
   double range, * values;
   double alpha, sigma;
   Covariance (double alp=-1, double sig = 1);
   Covariance (int nBin, double rang, double * vals)
      {type = 1; nBins = nBin; range = rang; vals = values;};
   double operator ()(double x);
};

class  A {
private:
   double (A:: *measure) (double x);
   double pureMeasure (double x);
   double convol   (double x);
public:
   int i, j;
   int nRegions;
   Border * border;
   double * areas;
   double SCALING; // The constant (Sum(areas)^(3/2)) to keep the
                   // weights (array Values) invariant to the scaling
                   // To get unscaled version multiply Values by SCALING.
   int minX, minY, maxX, maxY;

   double ** Values;
   double ** xes, ** wus;
   double diam;
   int nCells;

    Covariance * gamma;



   A (char * fname, int pure = 1, int preCalculate = 1);
   void writeW (char * fname);
   void readW (char * fname);

   double kmeasure (double x) {return (this ->*measure) (x);};

   double qgaus (double a, double b, int n = 30);
   double qfast (void);
};

Point Range (Border& a, Border& b);
```

```
#endif
```

C.4 kmeasure.cc

```
#include <stream.h>
#include <math.h>
#include <float.h>
#include "kmeasure.h"

/* A measure of a segment l
 *    intersecting parallel segments l1 and l2
 *    which are distance d apart and shifted by s
 */
double parallelM (double l, double l1, double l2, double d, double s)
{
    if (l <= d || l1 == 0 || l2 == 0)
        return 0;
    if (d == 0){
        double x = dmin (l1, s+l2) - dmax (0, s);
        if (x <= 0)
            return 0;
        else
            return 4 * x * l;
    }
    if (l1 > l2){// make sure l1 < l2
        double tmp = l2;
        s = l2 + s - l1;
        l2 = l1;
        l1 = tmp;
    }
    if (s + l2/2 < l/2)// Make sure the middle of l2 is to the right
        // of the middle of l1
        s = (-s + l1) - l2;

    double phi0 = asin (d/l);
    double phi1 = M_PI - phi0;
```

```
  double phiA, phiB, phi0MinAB, phiMaxAB1;
  double ans = 0;
  phi0MinAB = M_PI/2 − atan2 (s + l2, d);
  phiMaxAB1 = M_PI/2 − atan2 (s − l1, d);


  if (phiMaxAB1 <= phi0 || phi0MinAB >= phi1)
    return 0;
  phiA = M_PI/2 − atan2 (s, d);
  phiB = M_PI/2 − atan2 (s + l2 − l1, d);

  phi0 = dmax (phi0, phi0MinAB);
  phi1 = dmin (phi1, phiMaxAB1);

  double cosphi0 = cos (phi0);
  double sinphi0 = sin (phi0);
  double cosphi1 = cos (phi1);
  double sinphi1 = sin (phi1);
  if (phi0 > phiA){
//readlibreadlib(C):
//intint((l*sin(p)-d)*(l*cos(p)-s1),p=phi0..phi1);
    double s1 = s−l1;
    ans = −l*l*cosphi1*cosphi1/2 +
      l*s1*cosphi1−d*l*sinphi1 + d*s1*phi1 +
      l*l*cosphi0*cosphi0/2 − l*s1*cosphi0 +
      d*l*sinphi0 − d*s1*phi0;
  }else{// phi0 <= phiA
    if (phi0 < phiB){
      double s2 = s+l2;
      double cosphiB = cos (phiB);
      double sinphiB = sin (phiB);
      //intint((l*sin(p)-d)*(s2-d*cot(p)),p=phi0..phiB);
      ans = −l*s2*cosphiB − d*l*sinphiB
        − d*s2*phiB + d*d*log(sinphiB)
        + l*s2*cosphi0 + d*l*sinphi0
        + d*s2*phi0 − d*d*log(sinphi0);
      phi0 = phiB;
      cosphi0 = cosphiB;
      sinphi0 = sinphiB;
    }// now phi0 >= phiB;
    //intint((l*sin(p)-d)*l1,p=phi0..phiA);
```

```
        double cosphiA = cos (phiA);
        double sinphiA = sin (phiA);
        ans += −(l∗cosphiA+d∗phiA)∗l1+(l∗cosphi0+d∗phi0)∗l1;
        double s1 = l1−s;
        //intint((l∗sin(p)-d)∗(s1+d∗cot(p)),p=phiA. .phi1);
        ans += −l∗s1∗cosphi1 + d∗l∗sinphi1
            − d∗s1∗phi1 − d∗d∗log(sinphi1)
            + l∗s1∗cosphiA − d∗l∗sinphiA + d∗s1∗phiA
            + d∗d∗log(sinphiA);
    }

    if (ans < 0){
        cerr << l << " " << l1 << " " << l2 << " " << d << " " << s;
        cerr << " Less than 0 in parallelM " << ans << "\n";
        ans = 0;
    }
    return 2∗ans;
}

static inline double closedForm (double l,  double alpha, double li, double phi)
{
/∗      return 2∗l∗li ∗ (1+cos (phi)) - l∗l∗ (sin (phi) ∗ sin (phi)/2 +
        cos (alpha) / sin (alpha) ∗ sin (2∗phi) / 4) +
        (li∗li ∗ sin (2∗alpha) + l∗l ∗ cos (alpha) / sin (alpha)) ∗ phi / 2 +
        li∗li ∗ sin (alpha) ∗ sin (alpha) ∗ log (sin (phi - alpha));
∗/
    double cosP = cos (phi);
    double sinP = sin (phi);
    double cosA = cos (alpha);
    double sinA = sin (alpha);
    double ctgA = cosA/sinA;
    double ll = l∗l;
    double lili = li∗li;

    if (sinP∗cosA − sinA∗cosP < DBL_EPSILON){
        return 0;
    }
    return 2∗l∗li ∗ (1+cosP) − ll ∗ (sinP + ctgA∗cosP)∗sinP/2
        + (lili ∗ sinA∗cosA ∗ 2 + ll ∗ ctgA) ∗ phi / 2
        + lili ∗ sinA∗sinA ∗ log (sinP∗cosA − sinA∗cosP);
}
```

```
/* A measure of a segment l intersecting two rays
*      starting at the same point with an angle alpha
*/
double fullM (double l, double alpha)
{
    double sinA = sin (alpha);
    double cosA = cos (alpha);
    return l * l * (1 + (M_PI − alpha) * cosA / sinA) / 2;
}

/* The same as fullM, but we remove pieces of each ray of length
*      l1 and l2 correspondingly. The removed pieces start at the origin.
*/
double OutMeasure (double l, double l1, double l2, double alpha)
{
    if ( (l1 + l2)/l < DBL_EPSILON && alpha > DBL_EPSILON) {
        return fullM (l, alpha);
    }
    double cosA = cos (alpha);
    double sinA = sin (alpha);
    double phi10, phi11, phi20, phi21;
    double FirstTerm = 0, SecondTerm = 0;
    int noSecondTerm = 0;
    if (cosA > 0){ // alpha < Pi/2
        if (l2 * sinA >= l || l1 * sinA >= l){
            return 0;
        }else{
            phi10 = asin (l2/l*sinA) + alpha;
            if (l1 > l2 * cosA){
                phi11 = alpha + atan2 (l2 * sinA, l1 − l2 * cosA);
            }else{
                if (l1 < l2 * cosA − l*sin(M_PI/2 − phi10 + alpha)){
                    phi11 = M_PI − phi10 + alpha + alpha;
                    noSecondTerm=1;
                }else{
                    phi11 = M_PI + alpha − atan2 (l2 * sinA, l2 * cosA − l1);
                }
            }
            if (phi11 > phi10 + FLT_EPSILON)
                FirstTerm = closedForm (l, alpha, l2, phi11) −
                closedForm (l, alpha, l2, phi10);
            if (!noSecondTerm){
                phi20 = asin (l1/l*sinA) + alpha;

                if (l2 > l1 * cosA){
                    phi21 = alpha + atan2 (l1 * sinA, l2 - l1 * cosA);
                }else{
                    if (l2 < l1 * cosA - l*sin(M_PI/2 - phi20 + alpha)){
                        phi21 = M_PI - phi20 + alpha + alpha;
                    }else{
                        phi21 = M_PI + alpha - atan2 (l1 * sinA, l1 * cosA - l2);
                    }
                }
                if (phi21 > phi20 + FLT_EPSILON)
                    SecondTerm =  closedForm (l, alpha, l1, phi21)-
                            closedForm (l, alpha, l1, phi20);
            }
        }
    }else { // alpha > Pi/2
        if (l2 >= l || l1 >= l){
            return 0;
        }else{
            phi10 = alpha + asin (l2 * sin (M_PI - alpha)/l);
            phi11 = alpha + atan2 (l2 * sinA, l1 - l2 * cosA);
            if (phi11 > phi10 + FLT_EPSILON)
                FirstTerm = closedForm (l, alpha, l2, phi11) -
                    closedForm (l, alpha, l2, phi10);
            else
                FirstTerm = 0;
            phi20 = alpha + asin (l1 * sin (M_PI - alpha)/l);
            phi21 = M_PI + alpha - phi11;
            if (phi21 > phi20 + FLT_EPSILON)
                SecondTerm = closedForm (l, alpha, l1, phi21) -
                    closedForm (l, alpha, l1, phi20);
            else
                SecondTerm = 0;
        }
    }
    if (SecondTerm < 0 || FirstTerm < 0){
        cerr << l << " " << l1 << " " << l2 << " "
            << alpha << " " << FirstTerm << " " << SecondTerm << " Less than 0\n";
        if (SecondTerm < 0)
            SecondTerm = 0;
        if (FirstTerm < 0)
            FirstTerm = 0;
    }
    return FirstTerm + SecondTerm;
}
```

C.5 geom.cc

```
#include <stream.h>
#include <stdio.h>
#include <math.h>
#include <float.h>

#include "geom.h"


double distance (intPoint& a, intPoint& b)
{
    double tmp = (a .x − b .x)∗(a .x − b .x) +
        (a .y − b .y) ∗ (a .y − b .y);
    return sqrt (tmp);
}

// inner distance between point t and segment from-to
double innerDistance2 (intPoint& t, intPoint& from, intPoint& to)
{
    if (distance2 (to, from) < DBL_EPSILON)
        return distance2 (t, to);
    double T = (t .x − from .x)∗(to .x − from .x) +
        (t .y − from .y) ∗ (to .y − from .y);
    if (T < 0)
        return distance2 (from, t);
    else{
        double T = (to .x − t .x)∗(to .x − from .x) +
            (to .y − t .y)∗(to .y − from .y);
        if (T < 0)
            return distance2 (to, t);
        else{
            double a2 = ((t .y − from .y)∗(to .x − from .x)−
                                 (t .x − from .x)∗(to .y − from .y));
            double tmp = fabs (a2) / distance2 (to, from);
            return tmp;
        }
    }
}

// Inner distance between curve a and curve b
double innerDistance (int na, int nb, intPoint ∗ a, intPoint ∗ b)
{
    double dist = distance2 (a [0], b [0]);
    for (int i = 0; i < na; i++)
        for (int j = 0; j < nb; j++){
            double tmp = innerDistance2 (a [i], b [j], b [(j+1)%nb]);
            dist = dmin (dist, tmp);
        }
    for (i = 0; i < nb; i++)
        for (int j = 0; j < na; j++){
            double tmp = innerDistance2 (b [i], a [j], a [(j+1)%na]);
            dist = dmin (dist, tmp);
        }
    return sqrt (dist);
}

double Line::distance (Point& to)
{
    double den = sqrt (a∗a + b∗b);
    if (b != 0)
        return (to .y ∗ fabs (b) + a ∗ to .x  + c)/den;
    return (to .x ∗ fabs (a) + b ∗ to .y  + c)/den;
}

double Segment::innerDistance (Point& t)
{
    double T = (t .x − from .x)∗(to .x − from .x) +
        (t .y − from .y)∗(to .y − from .y);
    if (T < 0)
```

```
        return sqrt ((from - t) .norm ());
    else{
        double T = (to .x - t .x)*(to .x - from .x) +
            (to .y - t .y)*(to .y - from .y);
        if (T < 0)
            return sqrt ((to - t) .norm ());
        else{
            double a2 = ((t .y - from .y)*(to .x - from .x)-
                            (t .x - from .x)*(to .y - from .y));
            return sqrt (a2 * a2 / (to - from) .norm ());
        }
    }
}

double Segment::outerDistance (Point& t)
{

    double a1 = sqrt ((from - t) .norm ());
    double a2 = sqrt ((to - t) .norm ());
    return MAX (a1, a2);
}

double Segment::outerDistance (Segment& t)
{

    double a1 = outerDistance (t .from);
    double a2 = outerDistance (t .to);
    return MAX (a1, a2);
}

double Segment::innerDistance (Segment& t)
{
    double a1 = innerDistance (t .from);
    double a2 = innerDistance (t .to);
    double a3 = t .innerDistance (from);
    double a4 = t .innerDistance (to);
    return MIN(MIN (a1, a2), MIN (a3, a4));
}

/* calculates counter-clockwise angle from the second
    vector to the first vector
```

```
*/
extern "C" double leftAngle (intPoint& a, intPoint& b)
{
    double phi1 = atan2 ( a .y, a .x),
        phi2 = atan2 (b .y, b .x), pi2 = 2 * M_PI;
    if (phi1 < 0)
        phi1 += pi2;
    if (phi2 < 0)
        phi2 += pi2;
    if (phi1 < phi2)
        return pi2 + (phi1 - phi2);
    else
        return phi1 - phi2;
}

extern "C" int isClockwise (int NPoints, intPoint * p)
{
    double area = 0.0;
    int i;
    for (i = 0; i < NPoints - 1; i++)
        area += (p [i] .x * p [i + 1] .y) -
            (p [i + 1] .x * p [i] .y);
    area += (p [NPoints - 1] .x * p [0] .y) - (p [0] .x * p [NPoints - 1] .y);
    area /= 2.0;
    if (area > 0)
        return 0;
    else
        return 1;
}

extern "C" double area (int NPoints, intPoint * p)
{
    double area = 0.0;
    int i;
    for (i = 0; i < NPoints - 1; i++)
        area += (p [i] .x * p [i + 1] .y) -
            (p [i + 1] .x * p [i] .y);
    area += (p [NPoints - 1] .x * p [0] .y) - (p [0] .x * p [NPoints - 1] .y);
    return fabs (area / 2.0);
}
```

```
extern "C" double perimeter (int NPoints, intPoint * p)
{
    double perimeter = 0.0;
    int i;
    for (i = 0; i < NPoints-1; i++){
        double dx = p [i] .x - p [(i + 1)] .x;
        double dy = p [i] .y - p [(i + 1)] .y;
        perimeter += sqrt (dx*dx + dy*dy);
    }
    double dx = p [NPoints-1] .x - p [0] .x;
    double dy = p [NPoints-1] .y - p [0] .y;
    perimeter += sqrt (dx*dx + dy*dy);
    return perimeter;
}

// Calculate line (ax + by + c = 0) going through two distinct points
Line::Line (Point& p1, Point& p2)
{
    a = p1 .y - p2 .y;
    b = p2 .x - p1 .x;
    c = p1 .x * p2 .y - p2 .x * p1 .y;
}

// Check if two lines are parralel
int Line::parallel (Line& l2)
{
    double s = (a * l2 .b - b * l2 .a);
    if (s == 0)
        return 1;
    else
        return 0;
}

// Calculate Intersection of two lines
Point Line::intersect (Line& l2)
{
    double den = a * l2 .b - l2 .a * b;
    return Point ( (b * l2 .c - l2 .b * c)/den,
                   (c * l2 .a - l2 .c * a)/den);
}
```

```
// Check if collinear point O is between a and b
int Point::between (Point& a, Point& b)
{
    if ((*this − a) .norm () + (*this − b) .norm () < (a−b) .norm ())
        return 1;
    else
        return 0;
}

// Calculate measure of a segment of length l intersecting segments
// a and b simultaneously in a simple case (when a \cap b is not
// inside open a or inside open b.
static double KMeasureSimple (double l, Point& O, Segment& a, Segment& b)
{
    Point a1 = a .from, b1 = a .to, a2 = b .from, b2 = b .to;
    typedef enum {IN=−1, OUT=1} direction;
    direction dirA = OUT, dirB = OUT;
    if (b1 .between (O, a1) ||
          (fabs (b1 .x − O .x) < DBL_EPSILON &&
            fabs (b1 .y − O .y) < DBL_EPSILON)){
        dirA = IN;
        b1 = a .from;
        a1 = a .to;
    }
    if (b2 .between (O, a2)||
          (fabs (b2 .x − O .x) < DBL_EPSILON &&
            fabs (b2 .y − O .y) < DBL_EPSILON)){
        dirB = IN;
        b2 = b .from;
        a2 = b .to;
    }
    double l11 = sqrt ((O−a1) .norm ());
    double l21 = sqrt ((O−a2) .norm ());
    double l12 = sqrt ((O−b1) .norm ());
    double l22 = sqrt ((O−b2) .norm ());
    double l32 = ((b1−b2) .norm ());
    double l13 = sqrt ((O−a1) .norm ());
    double l23 = sqrt ((O−b2) .norm ());
    double l14 = sqrt ((O−b1) .norm ());
    double l24 = sqrt ((O−a2) .norm ());
    double cosA = ( (l12/l22 + l22/l12 - l32/l12/l22)/2 );
    double alpha = acos (cosA);

    return dirA*dirB * (-1 * OutMeasure (l, l11, l21, alpha) -
                             OutMeasure (l, l12, l22, alpha) +
                             OutMeasure (l, l13, l23, alpha) +
                             OutMeasure (l, l14, l24, alpha));
}

/* Calculate Kinematic measure of the movements of a segment of length l
    intersecting two other segments
*/
double KMeasureSegment (double l, Segment& a, Segment& b)
{
    if ((a .to == a .from) || (b .to == b .from))
        return 0;
    Line la = Line (a .from, a .to), lb = Line (b .from, b .to);
    if (la .parallel (lb)){
        double num = lb .a * a .from .x + lb .b * a .from .y + lb .c;
        double den = (lb .a*lb .a + lb .b*lb .b);
        double d = fabs (num) / sqrt (den);
        double l1 = sqrt ((a .to - a .from) .norm ());
        double l2 = sqrt ((b .to - b .from) .norm ());
        Point close (a .from .x - lb .a * num / den,
                     a .from .y - lb .b * num / den);
        Point dir = a .to - a .from;
        double sign, s;
        if ( (b .to - close)*dir < (b .from - close)*dir ){
            sign = 1;
            s = sqrt ((b .to - close) .norm ());
            if ((b .to - close)*dir < 0)
                s = -s;
        }else{
            sign = -1;
            s = sqrt ((b .from - close) .norm ());
            if ((b .from - close)*dir < 0)
                s = -s;
        }
        return sign * parallelM (l, l1, l2, d, s);
    }else{
        Point O = la .intersect (lb);
        if (O .between (a .from, a .to)){
            Segment x1 (a .from, O), x2 (O, a .to);
            return KMeasureSimple (l, O, b, x1) +
                    KMeasureSimple (l, O, b, x2);
        }else{
            if (O .between (b .from, b .to)){
                Segment x1 (b .from, O), x2 (O, b .to);
                 return KMeasureSimple (l, O, a, x1) +
                    KMeasureSimple (l, O, a, x2);
            }else{
                return KMeasureSimple (l, O, a, b);
            }
        }
    }
    return 0;
}




/∗ Calculate Kinematic measure of the movements of an oriented segment of
∗     length l so that it starts and ends within nonintersecting
∗     polygons a and b accordingly.
∗     All Polygons must have boundary vertices in same orientation!!!
∗/
extern "C" double KMeasure (double l, int an, int bn,
                                    intPoint ∗ a, intPoint ∗ b)
{
    if (l == 0)
        return 0;
    double sum = 0;
    if (a == b && an == bn){// The same polygon see Theorem
        for (int i = 0; i < an − 1; i++)
            for (int j = i+1; j < an; j++)
                if (j != i){
                    double tmp = KMeasureSegment (l,
                                        Segment (a [i], a [(i+1)%an]),
                                                    Segment (b [j], b [(j+1)%bn]));
                    sum += tmp;
            }
        sum += 2 * (M_PI * area (an, a) − l * perimeter (an, a));
    }else{
        for (int i = 0; i < an; i++)
            for (int j = 0; j < bn; j++){
                double tmp = KMeasureSegment (l,
                                    Segment (a [i], a [(i+1)%an]),
                                            Segment (b [j], b [(j+1)%bn]));
                sum += tmp;
            }
        sum /= 2;
    }
    if (sum < 0)
        return 0;
    else
        return sum;
}
```

C.6 A.cc

```
#include <stream.h>
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#include "geom.h"
#include "A.h"

Point Range (Border& a, Border& b)
{
    double oD = 0, iD;
    for (int i = 0; i < a .nPoints; i++)
        for (int j = 0; j < b .nPoints; j++){
            double tmp = distance2 (a .line [i],            b .line [j]);
            oD = dmax (oD, tmp);
        }
    oD = sqrt (oD);

    if (&a == &b) // the same curve
        iD = 0;
    else
        iD = innerDistance (a .nPoints, b .nPoints, a .line, b .line);

    Point ans (iD, oD);
    return ans;
}

Covariance::Covariance (double alp=−1, double sig = 1)
{
    if (alp > 0 || sig <= 0){
        fprintf (stderr,
        "need nonpositive first parameter and positive second\n");
        exit (−1);
    }else{
        type = 0;
        sigma = sig;
        alpha = alp;
    }
}

double Covariance::operator ()(double x)
{
    switch (type){
    case 0:
        double tmp = sigma ∗ exp (alpha ∗ (fabs (x)));
        if (tmp < 0)
            return 0;
        else
            return tmp;
        break;
    case 1:
        int which = (int) (fabs(x) / range ∗ nBins);
        if (which >= nBins)
            return values [nBins − 1];
        else
            return values [which];
        break;
    default:
        fprintf (stderr,
        "covariance type is not set up properly \n");
        exit (−1);
        return 0;
    }
}


void A::writeW (char * name)
{
    FILE * out = fopen (name, "w");
    if (out == NULL){
        fprintf (stderr, "Can not open file %s fr writing\n", name);
        exit (−1);
    }
    fprintf (out, "%d\n%d\n", nRegions, nCells);
    fwrite (&diam, sizeof(diam), 1, out);
    for (int i = 0; i < nRegions; i++)
        for (int j = i; j < nRegions; j++){
            fwrite (xes [i*nRegions + j], sizeof(**xes), nCells, out);
            fwrite (wus [i*nRegions + j], sizeof(**wus), nCells, out);
            fwrite (Values [i*nRegions + j], sizeof(**Values), nCells, out);
            fprintf (stderr, "%d %d %lf::: ", i, j, SCALING);
//              int k;
//              for (k = 0; k < nCells; k++)
//                  fprintf (stderr, "%.13le ", xes [i*nRegions + j][k]);
//              fprintf (stderr, "\n");
//              for (k = 0; k < nCells; k++)
//                  fprintf (stderr, "%.13le ", wus [i*nRegions + j][k]);
//              fprintf (stderr, "\n");
//              for (k = 0; k < nCells; k++)
//                  fprintf (stderr, "%.13le ", Values [i*nRegions + j][k]);
//              fprintf (stderr, "\n");
        }
    fclose (out);
}
```

```
void A::readW (char * name)
{
    FILE * out = fopen (name, "r");
    if (out == NULL){
        fprintf (stderr, "Can not open file %s for reading\n", name);
        exit (−1);
    }
    fscanf (out, "%d\n%d", &nRegions, &nCells);
    if (fgetc (out) != '\n'){
        cerr << "Some problem reading the W file\n";
        exit (−1);
    }
    fread (&diam, sizeof(diam), 1, out);
    for (int i = 0; i < nRegions; i++)
        for (int j = i; j < nRegions; j++){
            fread (xes [i*nRegions + j], sizeof(**xes), nCells, out);
            fread (wus [i*nRegions + j], sizeof(**wus), nCells, out);
            fread (Values [i*nRegions + j], sizeof(**Values), nCells, out);
            Values [j*nRegions + i] = Values [i*nRegions + j];
        }
    fclose (out);
}

double A::convol (double x)
{
    double tmp1 = (*gamma) (x),
           tmp2 = pureMeasure (x);
    return tmp1*tmp2;
}

double A::pureMeasure (double x)
{
    return x * KMeasure (x, border [i] .nPoints,
                            border [j] .nPoints,
                            border [i] .line,
                            border [j] .line);
}

A::A (char * fileName, int pure, int preCalculate)
{
    int i, j, size;
```

```
FILE * in = fopen (fileName, "r");
if (in == NULL){
    fprintf (stderr, "can not open file %s\n", fileName);
    exit (−1);
}
fscanf (in, "%d", &nRegions);
if ((border = (Border *) malloc (nRegions * sizeof (Border))) == NULL){
    fprintf (stderr, "Can not allocate memory\n");
    exit (−1);
}

for (i = 0; i < nRegions; i++){
    if (1 != fscanf (in, "%d", &size)){
        fprintf (stderr, "can not read file %s\n", fileName);
        exit (−1);
    }
    border [i] . nPoints = size;
    if ((border [i] .line = (intPoint *) malloc (size * sizeof (intPoint))) == NULL){
        fprintf (stderr, "can not allocate memory\n");
        exit (−1);
    }
    for (j = 0; j < size; j++)
        if (2 != fscanf (in, "%d %d", &(border [i] .line [j] .x),
                                  &(border [i] .line [j] .y))){
            fprintf (stderr, "can not read file %s\n", fileName);
            exit (−1);
        }
    if (preCalculate){
        border [i] .minX = border [i] .maxX = border [i] .line [0] .x;
        border [i] .minY = border [i] .maxY = border [i] .line [0] .y;
        for (j = 1; j < size; j++){
            if (border [i] .minX > border [i] .line [j] .x)
                border [i] .minX = border [i] .line [j] .x;
            if (border [i] .maxX < border [i] .line [j] .x)
                border [i] .maxX = border [i] .line [j] .x;
            if (border [i] .minY > border [i] .line [j] .y)
                border [i] .minY = border [i] .line [j] .y;
            if (border [i] .maxY < border [i] .line [j] .y)
                border [i] .maxY = border [i] .line [j] .y;
        }
    }
```

```
}
fclose (in);
if (preCalculate){
    minX = border [0] .minX;
    maxX = border [0] .maxX;
    minY = border [0] .minY;
    maxY = border [0] .maxY;
    for (i = 1; i < nRegions; i++){
        if (border [i] .minX < minX)
            minX = border [i] .minX;
        if (border [i] .maxX > maxX)
            maxX = border [i] .maxX;
        if (border [i] .minY < minY)
            minY = border [i] .minY;
        if (border [i] .maxY > maxY)
            maxY = border [i] .maxY;
    }
    diam = sqrt ((maxX − minX)∗(maxX − minX) +
                  (maxY − minY)∗(maxY − minY));
}
areas = new double [nRegions];
for (i = 0; i < nRegions; i++)
    areas [i] = ::area (border [i] .nPoints, border [i] .line);
SCALING = 0;
for (i = 0; i < nRegions; i++)
    SCALING += areas [i];
SCALING ∗= sqrt(SCALING);

nCells = 81;
Values = new double ∗ [nRegions ∗ nRegions];
xes = new double  ∗ [nRegions ∗ nRegions];
wus = new double ∗ [nRegions ∗ nRegions];
for (i = 0; i < nRegions; i++)
    for (j = i; j < nRegions; j++){
        Values [i∗nRegions + j] = new double [nCells];
        Values [j∗nRegions + i] = Values [i∗nRegions + j];
        xes [i∗nRegions + j] = new double [nCells];
        xes [j∗nRegions + i] = xes [i∗nRegions + j];
        wus [i∗nRegions + j] = new double [nCells];
        wus [j∗nRegions + i] = wus [i∗nRegions + j];
        if (preCalculate){
```

```
                    Point range = Range (border [i], border [j]);
                    quadrature (range .x, range .y, xes [i*nRegions + j],
                            wus [i*nRegions + j], nCells);
/*                 {// uniform spacing
                    for (int k = 0; k < nCells; k++){
                         wus [i*nRegions + j] [k] = (range .y-range .x)/nCells;
                         xes [i*nRegions + j] [k] =
                              range .x + (k+.5)*(range .y-range .x)/nCells;
                    }
               }
*/
                    this ->i = i; this ->j = j;
                    for (int k = 0; k < nCells; k++){
                         Values [i*nRegions + j] [k] =
                              pureMeasure (xes [i*nRegions + j][k])/SCALING;
//                           cout << i << " " << j << " " << xes [i*nRegions + j][k]
//                           << " " << wus [i*nRegions + j][k]
//                                 << " " << Values [i * nRegions + j][k]
//                                    << " " << "+-\n";
                    }
               }

          }

     if (pure)
          measure = &A::pureMeasure;
     else{
          gamma = new Covariance (−1, 1);
          measure = &A::convol;
     }
}

extern "C" void gaussq (int * kind, int * n, double * alpha, double * beta,
                      int * kpts, double * endpts, double * scratch,
                      double * x, double * w);
void quadrature (double x1, double x2, double *x, double *w, int n)
{
     // Legendre quadrature (kind = 1)
     // w(x) = 1 on (-1, 1)
     double * scratch = NULL, * X = NULL, * W = NULL, endpts [2] = { −1, 1 },
          zero = 0.0;
```

```
    int kind = 1, kpts = 2;
    static int N = 0;
    if (!N){
        X = (double *) malloc (n * sizeof (double));
        W = (double *) malloc (n * sizeof (double));
        scratch = (double *) malloc (n * sizeof (double));
        if (X == NULL || W == NULL || scratch == NULL){
            fprintf (stderr, "Can not allocate 3*%d doubles\n", n);
            exit (-1);
        }
        N = n;
        gaussq (&kind, &N, &zero, &zero, &kpts, endpts,  scratch,
                   X, W);
    }
    if (N != n){
        X = (double *) realloc (X, n * sizeof (double));
        W = (double *) realloc (W, n * sizeof (double));
        scratch = (double *) realloc (scratch, n * sizeof (double));
        if (X == NULL || W == NULL || scratch == NULL){
            fprintf (stderr, "Can not reallocate 3*%d doubles\n", n);
            exit (-1);
        }
        N = n;
        gaussq (&kind, &N, &zero, &zero, &kpts, endpts,  scratch,
               X, W);
    }

    for (int i = 0; i < n; i++){
        x [i] = (x2 - x1) * (X [i] + 1)/2 + x1;
        w [i] = W [i] * (x2 - x1)/2;
    }
}

double A::qgaus (double a, double b, int n)
{
    double xx [n], w [n];
    double sum = 0;
    quadrature (a, b, xx, w, n);
    for (int ii = 0; ii < n; ii++){
        sum += w [ii] * ((this ->*measure) (xx [ii]));
    }
    return sum;
}

double A::qfast (void)
{
    double tmp = 0;
    for (int k = 0; k < nCells; k++){
        tmp += Values [i * nRegions + j][k]
            * this ->gamma ->operator () (xes [i * nRegions + j][k])
            * wus [i * nRegions + j][k];
    }
    return tmp;
}
```