Efficient Global Optimization: Analysis, Generalizations and Extensions

Theresia Mayer

Doctor of Philosophy
University of Edinburgh
2003

Abstract

In some optimization problems the evaluation of the objective function is very expensive. It is therefore desirable to find the global optimum of the function with only comparatively few function evaluations. Because of the expense of evaluations it is justified to put significant effort into finding good sample points and using all the available information about the objective function. One way of achieving this is by assuming that the function can be modelled as a stochastic process and fitting a response surface or surrogate function to it, based on function evaluations at a set of points determined by an initial design. Parameters in the model are estimated when fitting the response surface to the available data. In determining the next point at which to evaluate the objective function, a balance must be struck between local search and global search. Local search in a neighbourhood of the minimum of the approximating function has the aim of finding a point with improved objective value. The aim of global search is to improve the approximation by maximizing an error function which reflects the uncertainty in the approximating function. Such a balance is achieved by using the expected improvement criterion. In this approach the next sample point is chosen where the expected improvement is maximized. The expected improvement at any point in the range reflects the expected amount of improvement of the approximating function beyond a target value (usually the best function value found up to this point) at that point, taking into account the uncertainty in the approximating function.

In this thesis, we present and examine the expected improvement approach and the maximization of the expected improvement function. Sometimes the data are deceptive and parameters estimated from the data can give misleading approximating and error functions. We discuss a variation of the expected improvement criterion which takes into account the potential error in the estimated parameters, and we consider a different stopping rule. We also investigate how the availability of derivatives may be exploited and the use of a non-constant regression function in the approximating function. As test functions we mainly use our own functions. These are sample paths of stochastic processes generated for this purpose. The generating process of these functions is also described.

Acknowledgements

First of all I wish to thank my Supervisors Dr. Julian Hall and Professor Ken McKinnon for their valuable help and guidance in carrying out the work presented in this Thesis, and I wish to thank Professor Sandy Davie for many helpful comments and discussions.

Secondly, I would like to thank my friends for their support, in particular my two officemates Jeremy Brookman and Jörg Sixt. We had many interesting and helpful discussions, and in stressful times the hot chocolates together worked wonders. One of the first people I got to know in Edinburgh is Dr. Andreas Grothey, I would like to thank him for his help throughout my time here. I would also like to thank Dr. Yoshiro Shibasaki for the support and encouragement he has given me.

Throughout this sometimes difficult time, my family, and my parents in particular, have always been very supportive and generous. I cannot thank them enough for all their help and their patience, particularly when nursing me back to health after my operations, and in the final stages of the course of this degree.

Declaration

I declare that this thesis was composed by myself and that the work contained therein is my own, except where explicitly stated otherwise in the text.

(*Theresia Mayer*)

Table of Contents

List of Figures

List of Tables

Notation

$\boldsymbol{x}$	a vector
$\boldsymbol{x}^{(i)}$	the ith vector
x_i	the ith entry of the vector $\boldsymbol{x}$
$\boldsymbol{e}^{(i)}$	vector with 1 in position i and all other entries 0
$\mathbf{1}$	vector with all entries 1
$\mathbf{0}$	vector with all entries 0
I	identity matrix
$\hat{y}$	an estimate of y
$y(\boldsymbol{x})$	objective function
y_{0n}	best (i.e. least) of the first n values of $y(\boldsymbol{x})$
d	dimension
D	sample space, feasible region
R	correlation matrix
$\boldsymbol{r}$	correlation vector
F	matrix of regression functions
$\boldsymbol{f}$	vector of regression functions
cov	covariance
c	autocovariance function
c_D	autocovariance function of discretely generated process
c_G	exponential autocovariance function
cor	correlation
ρ	autocorrelation function
ρ_G	exponential autocorrelation function
$\tilde{c}$	Fourier transform of c

Chapter 1

Introduction

In many applications, for example in industrial design and engineering problems, the evaluation of the objective function can take a long time, and in that sense be very expensive. The expense of function evaluations makes it particularly desirable to find the global optimum of the objective function with as few function evaluations as possible. Typically, such optimization problems from industry have only a small number of variables, but often little or nothing is known about the mathematical structure of the problem. This can make the choice of optimization method and the setting of the parameters of the method difficult, and a good choice often relies on the experience of the user.

The optimization problem considered in this thesis is of the form

$$\min\{y(\boldsymbol{x})|\boldsymbol{x} \in D = [0,1]^d\}.$$

By scaling the variables, a general box constrained optimization problem can be written in this form. Throughout we assume deterministic functions, i.e. repeated evaluations at the same point give the same value.

Problems where function evaluations are expensive are for example helicopter rotor blade design, investigated by Booker *et al.* in [8], or aerodynamic wing optimization, considered by Alexandrov *et al.* in [1]. Another example, used by Schonlau in [63] is piston design, where the aim is to design a piston for an automobile engine which has minimal undesirable motion and friction below a certain value. Other applications include thermal energy storage systems, where a 'utility index' depends on the freezing/melting temperature of the phase change material and the thickness of its layer; this problem is considered for example by Currin *et al.* in [15]. Function evaluations in these examples are expensive, since they involve the simulation of a realistic physical system.

When searching for the global optimum of any function a major difficulty lies in getting trapped at a stationary point or local minimizer. Different strategies have been suggested in an attempt to overcome this problem. Generally, most

global optimization methods have both local and global search strategies, but how to combine these strategies is a dilemma: a balance must be struck between "local refinement" and "global reliability". The global search strategy should ensure that the method does not get trapped at a local minimizer, the local strategy should ensure that regions of good minimizing points are investigated. The added difficulty of the expense of function evaluations means that the number of function evaluations should be kept low, and as much available information about the function as possible should be exploited in the optimization process. This can be done as follows: initially, the objective function is evaluated at a set of initial sample points, then in the following optimization process it is only evaluated at promising points in an attempt to keep the number of evaluations low. Promising candidate points are found by optimizing a merit function, which is itself a global optimization problem and requires some effort to solve. However, if objective function evaluations are expensive it is justified to put significant effort into finding good new candidate points at which to evaluate the function. Most merit functions used in practice are a combination of an approximating function, often called response surface or surrogate function, and an error function, the latter reflecting the uncertainty in the approximating function. Such functions thus incorporate both a local and a global search strategy for a good candidate point, with the aim of globally improving the approximating function while investigating the regions of its minima.

One type of approach to optimizing expensive functions is the often termed "random function" approach. This assumes that the objective function can be modelled by a realization of a stationary Gaussian stochastic process. This then allows for statistical interpretation, and the objective function value $y(\boldsymbol{x})$ at any unsampled point $\boldsymbol{x} \in D$ can be interpreted as a random variable $Y(\boldsymbol{x})$ with a certain distribution. This approach has entered global optimization from Kriging in Geostatistics. The term Kriging refers to an interpolation method, and derives from the name of a South African mining engineer D. G. Krige, who developed the method to predict ore reserves. The Kriging method is based on the assumption that points close together have a certain degree of spatial correlation, whereas points a large distance apart are statistically independent (cf. [21]). The data obtained from evaluating the function at a set of initial sample points are interpolated with a linear combination of basis functions and regression functions, to give a function which approximates the objective. This approximating function is used as a posterior mean. The basis functions have parameters that are tuned in the process of fitting the approximating function to the data. An error function reflects the uncertainty in the approximating function and, at any point, can

be interpreted as a variance of $Y(\boldsymbol{x})$, i.e. a measure of deviation from the mean value, given by the approximating function, at that point $\boldsymbol{x} \in D$. A combination of minimizing the approximating function and maximizing the error function therefore incorporates both a local and a global search strategy and could be used as a merit function to find new candidate sample points. One particular such merit function, the expected improvement function, and possible variants of it, are examined in detail in this thesis.

A review of global optimization methods in general is given in Chapter 2. Chapter 3 concentrates more specifically on the background of a certain type of method, the random function method, with a view on the particular search strategy for new sampling points which is investigated and of which new variants are presented later in this thesis, the expected improvement criterion. The necessary mathematical tools are given and the setting is described. In the approach explained in Chapter 3, it is assumed that the objective function can be modelled by a stationary Gaussian stochastic process. This means that the optimization method should be particularly well suited for sample paths of such processes when used as objective functions. The generation of such sample paths as test functions is described in Chapter 4. These test functions can easily be generated in large numbers and are useful when analyzing and comparing the global optimization methods investigated. Most of the test functions used as examples in the later chapters are of this kind. Some failure cases in the global optimization, and some possible improvements and extensions to the expected improvement method are examined in Chapter 5. Sometimes the optimization stops prematurely. This happens if the data are deceptive, the estimated parameters do not match the objective function well, or the model that is used for the objective function is not very accurate. Some ways of overcoming this problem by means of trying to overcome the misleading parameter values are investigated in Chapter 5. Also addressed are the issues of a good choice of number of initial sample points and the setting of a stopping tolerance, and some computational results are presented. The use of non-constant regression functions and derivatives is explained in Chapter 3 and the necessary theory developed. Chapter 6 examines the use of these in practice. Some computational results are presented for variants of the algorithm using quadratic regression functions and derivatives, respectively, and these are compared to results for variants using constant regression only and no derivatives. To conclude the presented work, Chapter 7 summarizes problems and achievements and gives pointers to possible future work.

The relevant implementations are in Fortran77 or Fortran90, using the Harwell Subroutine Library Release 12, and the NAG Fortran 77 Library Mark 18.

Chapter 2

Review

This chapter gives an overview of existing global optimization techniques. Section 2.1 concentrates on global optimization methods in general, and Section 2.2 deals in more detail with a certain kind of stochastic approach — methods using a response surface. We give more detailed reviews of methods which are more relevant to our work.

The optimization literature contains numerous accounts of global optimization techniques. A wide range of global optimization methods are summarized and explained for example in Törn and Žilinskas' book [71], and the contribution by Boender and Romeijn to the book [5]. Focusing on computational aspects and the methods being put to use in practice are for example Mongeau *et al.* in [49]; here different public-domain software products for global optimization of black box functions are compared for their efficiency and ease of use. In [32] Jones gives an overview of response surface methods in global optimization, which are the methods investigated in this thesis. Other, local optimization, methods are sometimes used as heuristics for global optimization. Some such local optimization methods are summarized and investigated for example by Powell in [57] and by Kolda *et al.* in [39].

2.1 Global Optimization Techniques

The aim of this section is to give an overview of global optimization techniques in general. It would be practically impossible to cover all existing methods and variants of methods, therefore we distinguish several larger classes of methods and give an introduction to each class and to representative or typical methods in each class. To be able to do this we need a classification of global optimization methods. Many different classifications have been proposed. To draw the line between two classes we need to take into consideration crucial differences between various methods. Pardalos *et al.* in [53] state that the "the major difference

between optimization problems is based on the presence or absence of convexity" but this is immediately followed by the comment that "in most optimization problems convexity of the objective function or the feasible domain is not easily recognizable". In their paper about recent developments in the field of global optimization [53], Pardalos *et al.* divide methods into *deterministic approaches* and *stochastic approaches*, and subclasses of these. Very generally, *deterministic approaches* exploit known analytical properties of the optimization problem. *Stochastic approaches* are methods for which the outcome in some sense is random. These are suitable for problems where nothing is known about analytical properties or structure.

Törn and Žilinskas in [71] suggest dividing the methods into *methods with guaranteed accuracy*, which implies exhaustive search for the global optimum in the region over which the function is to be optimized, and the remaining methods are divided into two classes, *direct methods*, and *indirect methods*. The two latter classes are divided up further and the more refined classification looks as follows:

- Methods with guaranteed accuracy

  TZ.1 Covering Methods

- Direct methods

  TZ.2 Random search methods

  TZ.3 Clustering methods

  TZ.4 Generalized descent methods

- Indirect methods

  TZ.5 Methods approximating the level sets

  TZ.6 Methods approximating the objective function.

Not all of these types of methods are relevant for our work. Therefore we will adapt our classification slightly for our purposes and only consider the kind of *indirect methods* which use approximations of the objective function, and split this up into *random function methods* and *trust region methods*. So the following modification of the above classification will be used, without any claim regarding completeness:

- Methods with guaranteed accuracy

  M.1 Covering Methods

- Direct methods

  M.2 Random search methods

  M.3 Clustering methods

- Indirect methods

  Methods approximating the objective function

  M.4 Random function methods

  M.5 Trust region methods.

We will now give a summary of certain types of global optimization methods, according to the above classification. Details of the methods can be found for example in Törn and Žilinskas' book [71], or in other references, as given in the relevant text passages.

Methods called *direct search methods* are presented in Powell [57] and Kolda *et al.* in [39]. Note however that these do not correspond to direct methods in the classification above. We explain the terminology and give a short summary of these methods in Section 2.1.4.

2.1.1 Methods with Guaranteed Accuracy

If the objective function has a known bounded rate of change, the global optimum of the function can be determined up to a prescribed accuracy by evaluation of the objective function at points close enough together, for example points on a dense enough grid. The construction of a grid is also called a covering and *methods with guaranteed accuracy* are often referred to as *covering methods.* It should be noted however, that it is often not possible to find a bound on the rate of change of the objective function. An estimate can be used, but then the method can no longer guarantee to find the global optimum.

2.1.1.1 Covering Methods

An example of a basic covering method is a search for the global optimum on a regular grid. The higher the density of the grid, the more likely it is that we find the global optimum of the objective function. Using a grid of high density requires the objective function to be evaluated at a large number of points. It would seem to make more sense and be more efficient to concentrate the search for the global optimum, and therefore the sample points, around promising areas rather than sampling the whole of D with the same density. This gives rise to *branch and*

bound type methods which subdivide the feasible region adaptively. At any stage in branch and bound a region is either excluded from further sampling because the bounds show that it cannot include the global optimum, or it is split into smaller subregions. By continuing this process the global optimum can be found to within any specified tolerance. For more details see for example Törn and Žilinskas' book [71].

2.1.2 Direct Methods

In *direct methods* local information, i.e. function values, is used, but no attempt is made to construct a model of the objective function from it. Boender and Romeijn's contribution to the book [5], and in Törn and Žilinskas' book [71].

2.1.2.1 Random Search Methods

The most basic algorithm of the type of a random search method is *pure random search.* In this method the objective function is evaluated at a number of randomly chosen points from a prespecified distribution over D. Often the uniform distribution is used. The best objective function value is used as an estimate of the global optimum. To have a good chance of finding the global optimum the objective function has to be evaluated at a large number of points.

Iterative variants are *pure adaptive search* and *adaptive search.* In *pure adaptive search* a new sample point is chosen from a (uniform) distribution on the set of points which improve the objective function value compared to previous sample points. In practice, however, these *improving regions* are difficult to construct. In *adaptive search* in every iteration the objective function is evaluated at a number of randomly chosen points from a distribution over D, and the distribution changes adaptively.

Another related approach is *simulated annealing.* This method avoids getting trapped at a local minimum by not only allowing iterations that lead to an improvement in the objective function value, but also, in a limited way, allowing iterations where the objective function value gets worse. As the algorithm progresses, the probability of accepting a deterioration in the objective function value decreases. This is called *cooling* and is done according to a *cooling schedule*, the choice of which can be a difficult issue.

Other quite simple *random search methods* include *singlestart* and *multistart* methods, also often called *two-phase methods.* Besides the random sampling *global phase*, these methods contain a phase of local refinement: the *local phase.* The function is initially evaluated at a number of randomly chosen points. This data is then used to find a candidate global optimum. In the *singlestart* and *multistart*

algorithms for example, an initial design is generated from a uniform distribution over D. In *singlestart* a single local search is started from the best point. In *multistart* a local search is started from each of these initial points (or a subset) to find different local optima. The best result of the local optimizations is used as an estimate of the global optimum.

Basic random search methods are quite popular because they are easily understood and implemented. But generally these methods require a large number of function evaluations in D and tend to be inefficient. This leads us on to the development of other methods. For example, several variants of *multistart* have been proposed and investigated, where an attempt is made to find every local optimum only once. *Clustering methods* are such methods.

2.1.2.2 Clustering Methods

Earlier publications on *clustering methods* include papers by Becker and Lago in 1970, and Törn in 1973. Essentially *clustering methods* are improved *random search methods*, and rely on a 'sensible' mixture of random sampling and local optimization. The aim is to prevent the algorithm from finding a local minimum more than once. If sample points can be clustered around local optima, and each cluster identifies the neighbourhood of a local optimum, such that the neighbourhoods of all local optima are captured in this way, then starting a local optimization from each cluster would give all the local optima, and therefore the global optimum. *Clustering methods* vary in the way the clusters are formed. Usually clusters are initiated by a *seedpoint* and other points are added to existing clusters in a stepwise fashion, depending on a distance criterion.

An algorithm for a *clustering method* starts by sampling an initial number of points in D. Then these points are pushed towards the local optima by performing a few steps of a local optimization starting from each of the points. Alternatively the points with a function value above a threshold could be discarded and only a predetermined number of points kept. Next, the points are grouped into clusters, and the resulting clusters are identified. Ideally, the points would now be in clusters around the local optima and all local optima would be captured in this way. If a stopping criterion is met, then, starting from the best point in each cluster, a local optimization is performed to determine the local optima. The best resulting local optimum is chosen as an estimate of the global optimum. If the stopping criterion is not met, we continue by for example treating points in better clusters (better in the sense of better objective function values at points in the cluster) as starting points for a new search. An alternative to using better clusters as new problems is retaining one point from every cluster and adding a

new set of randomly chosen points from a uniform distribution over D. Then the process is repeated.

Clustering methods, or techniques inspired by these, include *linkage* type algorithms. One such is the *random linkage* algorithm. Here a single point randomly chosen from the uniform distribution over D is sampled in each iteration. With a probability depending on the distance to the closest point with better objective function value, a local search is started from that point. Another example is *single linkage* where all points within a critical distance of points in a cluster are added to that cluster. For more details of this and other related algorithms see for example Locatelli and Schoen's work in [43].

2.1.3 Indirect Methods

Indirect methods use available local information to build an approximation of the objective function, which is used to guide the search for promising sample points. We classify such methods according to whether or not the model attempts to be globally valid, and distinguish here between *random function methods* and *trust region methods.*

2.1.3.1 Random Function Methods

This is the type of method we are most interested in, and will be most concerned with later on. These methods attempt to find a global minimum of the objective function by modelling it and taking into account the uncertainty of the model. We will only give a short introduction to the idea behind the methods here and will introduce some existing algorithms in more detail later on. Kushner in [40] was the first to investigate the random function approach in a global optimization setting in 1962. In 1964 Kushner proposed a global optimization algorithm using a statistical model (Wiener process) of the objective function, see [41]. Random function methods have since attracted a lot of attention and more recent developments will be summarized later on.

In this approach a function is assumed to be the sample path or realization of an a priori stochastic process. As a consequence, the unknown function values of the objective function are treated as random variables. Known objective function values at the sample points can be used to determine a posterior distribution of the objective function value at any point $\boldsymbol{x} \in D$. In this sense a conditional mean and variance give the expected function value at any point $\boldsymbol{x} \in D$ and how uncertain this approximation to the function is at the point $\boldsymbol{x}$. We will also refer to these methods as *response surface methods.* This kind of method stems from a Bayesian approach to global optimization. A prior (a priori) distribution

is fixed on a set of functions $y(\boldsymbol{x})$. This distribution is then updated to take into account observations $(\boldsymbol{x}^{(i)}, y_i)$, $i = 1, \ldots, n$ of the objective function, using laws of conditional probability. The updated distribution is called a posterior (a posteriori) distribution. The posterior distribution can be used to determine the next point of observation $\boldsymbol{x}^{(n+1)}$ by minimizing a risk function which reflects the expected deviation from the global minimum at any point in D. See for example the papers by Mockus [47, 48].

Other methods combining approximations of the objective function with *trust region methods* have been investigated. A brief introduction to some of these will be given now, before we describe other *response surface methods* in more detail later on.

2.1.3.2 Methods using Trust Regions

The use of *response surfaces* in conjunction with *trust regions* has been investigated by a number of people, see for example the work of Alexandrov *et al.* in [1] and Conn *et al.* in [10, 11, 12]. Generally, a model of the objective function is assumed to be valid within a certain distance of the current point: the *trust region.* The model is minimized to find new promising points, hopefully with better objective function value than the previous best value found. These methods do not model uncertainty about the objective function and only attempt to find local minima.

Alexandrov *et al.* in [1] investigate a *trust-region Approximation Management Framework.* AMF uses cheaper to compute lower-fidelity models in iterative procedures, and monitors the progress of the algorithm by occasionally returning to the higher-fidelity models. At the current point a response surface is used to find new promising points. The response surface is optimized within a prescribed maximum distance from the current point, a trust region. The new point can be used to update the model and optimize it again, or to serve as a new current point. Properties of the model are assessed and the model, the current point, and maximum distance are updated. Conn *et al.* in [10, 11, 12] propose approximating the objective by a quadratic model. The quadratic model is assumed to be a good approximation to the objective function within a region of a given radius around a point, the *current iterate*, see for example [12]. The model is minimized within this trust region. The actual objective function is evaluated at the optimum point and the reduction in the objective function and the model are compared. With a good ratio between these reductions, the trust region radius is increased. Then the model, the trust-region, and the current point are updated. An important difference between this approach and other trust-region methods

is the following: besides the interpolation conditions for the quadratic model m_k in iteration k, $m_k(\boldsymbol{x}^{(i)}) = y(\boldsymbol{x}^{(i)})$, $i = 1, \ldots, n$, some *geometric conditions* are imposed on the *interpolation set* $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}\}$ to "ensure existence and uniqueness of the quadratic interpolant" (Conn *et al.* [10] p. 6). This might lead to removal of points from the *interpolation set* in the course of the algorithm. It might also lead to finding new points with better function values at the stage in the algorithm where the function is evaluated at new points to improve the geometry of the interpolation set. For details see for example the paper by Conn *et al.* [10]. In [12], Conn *et al.* investigate an extension of this method to certain constrained optimization problems.

In 1969, Winfield used objective function values to build a quadratic interpolant of the objective function, which he then minimized within a trust region to find new promising points. For the main idea of Winfield's approach see Winfield's thesis [73]. Several years later, Powell proposed a related method: certain geometric properties of the interpolation set were preserved, avoiding some difficulties that were encountered in earlier methods. For more details of Powell's work see [55] and [56]. The work of Conn *et al.* is largely based on Powell's work.

2.1.4 Alternative Definition of Direct Search Methods

The term *direct search* is used differently by different authors. For example Powell in [57] defines a direct search method as one that only uses function values. However, these may or may not be used to build a model of the objective function. They therefore include both direct methods and indirect methods as defined above. The methods described by Powell are essentially local optimization methods. However because they sample the function at well spaced points they may avoid getting trapped in narrow local optima.

Generally *direct search methods* as defined by Powell are non-derivative methods and often used when derivatives of the objective function are not available. Several methods have been proposed, some with the aim to prove convergence of the method, some with the aim to make any improvement in the best function value. Methods considered by Powell in [57] which have not been mentioned here so far, are *line search methods*, *discrete grid methods*, *simplex methods*, *conjugate direction methods*, *linear approximation methods*, and *quadratic approximation methods*.

The idea of *line search methods* is that for a current iterate a direction is chosen, and the objective function is minimized along the line which passes through the current iterate and has the chosen direction. Extra rules can be imposed to help avoid bad behaviour like cycling. *Conjugate direction methods* are line search

methods, but the directions are required to satisfy a conjugacy criterion. These methods are originally designed to be efficient for unconstrained minimization when the objective function is a convex quadratic.

Simplex methods are based on the use of simplices. A *simplex* is the convex hull of $n + 1$ points in $\mathbb{R}^n$. At the beginning of each iteration such a simplex is available. The objective function values at the vertices of the simplex are used to determine a new simplex for the next iteration. Usually the vertex with the worst function value is reflected through the centroid of the other vertices, to give a new vertex, and therefore the new simplex. Extra rules must be imposed to avoid cycling. Under certain conditions a new simplex is obtained by shrinking the current simplex, rather than reflecting a vertex.

Discrete grid methods are suitable for bound constrained problems. A mesh of gridpoints is chosen, on which it is attempted to reduce the objective function value. The refinement of the grid can be altered as the algorithm proceeds.

The *linear approximation methods* and *quadratic approximation methods* are based on local approximations to the objective function by linear polynomials and quadratic polynomials, respectively. These approximations interpolate the objective in a certain number of points, for example the best points found. Minimizing the approximation gives new points at which the objective is evaluated. Under certain circumstances a *simplex step* is performed, which discards certain points and replaces them by new ones, to be then interpolated by the approximating linear or quadratic polynomial. These methods lead on to the kind of trust region method described earlier in this chapter.

2.1.5 Conclusions

Most of the summarized methods have been studied and used in practice. Some of the methods are appealing because they are easily understood and implemented, but many of them, in particular methods with guaranteed accuracy and direct methods, often require the objective function to be evaluated at a large number of points. This is prohibitive if function evaluations are expensive. In this situation indirect methods are more attractive.

2.2 Random Function Methods

In this section some recent developments in global optimization of expensive functions using stochastic approaches are introduced. The methods are based on the use of response surfaces to approximate the objective function. These are then used to choose new sample points and thus guide the optimization. The opti-

mization problem is of the form

$$\begin{aligned} \min \quad & y(\boldsymbol{x}) \\ s.t. \quad & \boldsymbol{x} \in D = [0,1]^d \end{aligned}$$

where $\boldsymbol{x}$ is a (scaled) input vector and the continuous function $y(\boldsymbol{x})$ is multimodal, in the sense of having several local optima, and expensive to evaluate. Note that we use the term response surface as an alternative name for a surrogate function.

These methods follow the same paradigm: the objective function is sampled at a number n of initially chosen sample points, also called an experimental design. A surface, called the response surface, is fitted to the sampled points and used as a predictor for the values of the function at unsampled points. A utility function based on this predictor is optimized to find the next sampling point $\boldsymbol{x}^{(n+1)}$. The predictor is then updated and the process repeated until a stopping criterion is met.

An account of such methods is given for example in Jones' paper [32], the outline of which will to some extent be followed here.

2.2.1 Interpolating Surfaces

Several methods that interpolate a set of scattered data have been investigated and described in the literature. These include for example cubic splines, thin-plate splines, and Kriging. Another approach, not interpolating the data, is to fit a surface to the function by least squares using polynomial terms and to minimize this surface to find a new sample point. A problem with this approach is that the response surface might not improve much as new sample points are added, and might fail to even find a local minimum, as Jones demonstrates in [32] for a quadratic surface. This problem can be avoided with interpolating surfaces, which have the desirable property that with every new sample point added the surface becomes more accurate. Minimizing such a surface to find a new sample point, updating it, and repeating, can be expected to lead to finding a local optimum of the objective function. As a way of interpolating the data we consider the basis function approach. Here the model consists of polynomial terms as well as basis function terms. Suppose we have sampled the function $y(\boldsymbol{x})$ at n points $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$ where $\boldsymbol{x}^{(i)}$ is a d-dimensional vector $\boldsymbol{x}^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^t$ and the functions values at these points are $y_i = y(\boldsymbol{x}^{(i)})$. Let $\{\pi_i(\boldsymbol{x}) | i = 1, \ldots, m\}$ be a basis of the space of all polynomials in $\boldsymbol{x}$ of degree g. The predictor $\hat{y}(\boldsymbol{x})$ is of the form

$$\hat{y}(\boldsymbol{x}) = \sum_{k=1}^{m} a_k \pi_k(\boldsymbol{x}) + \sum_{j=1}^{n} b_j \rho(\boldsymbol{x} - \boldsymbol{x}^{(j)}). \tag{2.1}$$

Possible basis functions are for example

$$\begin{aligned}\rho(\boldsymbol{z}) &= \|\boldsymbol{z}\|^3 \\ \rho(\boldsymbol{z}) &= \|\boldsymbol{z}\|^2 \log(\|\boldsymbol{z}\|) \\ \rho(\boldsymbol{z}) &= \exp\left(-\sum_{i=1}^{d} \theta_i |z_i|^{p_i}\right),\end{aligned}$$

where $\|z\|$ is the Euclidean norm. These are the basis functions used for cubic splines, thin-plate splines, and Kriging respectively. Some special cases of the Kriging basis functions are often considered separately: the exponential correlation functions where $p_i = 1$, $i = 1, \ldots, d$, and the Gaussian correlation functions where $p_i = 2$, $i = 1, \ldots, d$. Following for example Sacks *et al.* in [60] we focus on the Kriging approach where

$$\rho(\boldsymbol{z}) = \exp\left(-\sum_{k=1}^{d} \theta_k |z_k|^{p_k}\right) \tag{2.2}$$

is used, with parameters $\theta_i > 0$ and $0 < p_i \leq 2$. Kriging has a statistical interpretation. Not only can we compute the interpolator or predictor, which at each point can be interpreted as the expected function value, but also the error in the predictor. This will be zero at sample points and will rise between sample points. The function $\hat{y}$ in (2.1) has $n+m$ parameters and is required to interpolate n values. With basis functions of the form given, this can be done for example with a constant polynomial term, i.e. with $m = 1$ and $g = 0$.

2.2.2 Kriging and the BLUP

In Kriging we assume that the covariance and the correlation between two function values $Y(\boldsymbol{x})$ and $Y(\boldsymbol{z})$ depends only on the distance between the points $\boldsymbol{x}$ and $\boldsymbol{z}$, i.e. $\text{cov}(Y(\boldsymbol{x}), Y(\boldsymbol{z})) = c(\boldsymbol{x} - \boldsymbol{z})$ and $\text{cor}(Y(\boldsymbol{x}), Y(\boldsymbol{z})) = \rho(\boldsymbol{x} - \boldsymbol{z})$. As a surrogate of the objective function in the Kriging approach we need to calculate a predictor of the function, the best linear unbiased predictor (BLUP). We first describe the case when the polynomial term in the predictor is a constant, μ. This corresponds to (2.1) with $m = 1$, $\pi_1 = 1$, and $a_1 = \mu$. Later on, the polynomial terms will be introduced. The reason for this is that if the objective function can be approximated well by a polynomial, then fewer function evaluations will be needed if the predictor $\hat{y}$ contains polynomial terms. In the derivation of the best linear unbiased predictor (BLUP) we follow Jones' "gentle introduction to kriging" as described in [32]. Suppose we want to predict the value of a function at a point $\boldsymbol{x}$ in the domain. We can model our uncertainty about the value of the function at this point by assuming that it is the realization of a normally

distributed random variable $Y(\boldsymbol{x})$ with mean μ and variance σ^2. Consider two points $\boldsymbol{x}$ and $\boldsymbol{z}$. We are uncertain about the value of the function at these points before we have sampled there. But if the function is continuous, $y(\boldsymbol{x})$ and $y(\boldsymbol{z})$ will be close if the distance between the two points $\|\boldsymbol{x}-\boldsymbol{z}\|$ is small. Statistically we can say that $Y(\boldsymbol{x})$ and $Y(\boldsymbol{z})$ will be highly correlated if $\|\boldsymbol{x}-\boldsymbol{z}\|$ is small. The correlation, see Definition 3.1.2, is assumed to have the form

$$\begin{aligned}\rho_K(\boldsymbol{x}-\boldsymbol{z}) &= \operatorname{cor}_K[Y(\boldsymbol{x}),Y(\boldsymbol{z})] \\ &= \exp\left(-\sum_{k=1}^{d}\theta_k|x_k-z_k|^{p_k}\right)\end{aligned}$$

which corresponds to (2.2), with parameters $\theta_k > 0$ and $0 < p_k \leq 2$, which have to be estimated from the data. As before, assume that the function has been evaluated at points $\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(n)}$ and let the observed values be denoted by $\boldsymbol{y} = (y_1,\ldots,y_n)^t$. Let R denote the $n\times n$ matrix with entries $R_{ij} = \operatorname{cor}_K[Y(\boldsymbol{x}^{(i)}),Y(\boldsymbol{x}^{(j)})]$. This matrix R is positive definite, and therefore invertible. We can use the likelihood function

$$L(\boldsymbol{y}) = |2\pi\sigma^2 R|^{-1/2}\exp\left[\left(-\frac{1}{2}\right)(\boldsymbol{y}-\mathbf{1}\mu)^t\sigma^{-2}R^{-1}(\boldsymbol{y}-\mathbf{1}\mu)\right]$$

to find the maximum likelihood estimates of the parameters μ, σ, θ_k, and p_k. For practical convenience usually the log-likelihood is maximized, which is

$$\ln L(\boldsymbol{y}) = -\frac{1}{2}\ln\left(|2\pi\sigma^2 R|\right) - \frac{1}{2}(\boldsymbol{y}-\mathbf{1}\mu)^t\sigma^{-2}R^{-1}(\boldsymbol{y}-\mathbf{1}\mu). \tag{2.3}$$

Taking derivatives with respect to μ and σ and setting these to zero gives the maximum likelihood estimates (MLE) for μ and σ, they are

$$\hat{\mu} = \left(\mathbf{1}^t R^{-1}\mathbf{1}\right)^{-1}\mathbf{1}^t R^{-1}\boldsymbol{y} \tag{2.4}$$

$$\hat{\sigma}^2 = \frac{1}{n}(\boldsymbol{y}-\mathbf{1}\hat{\mu})^t R^{-1}(\boldsymbol{y}-\mathbf{1}\hat{\mu}). \tag{2.5}$$

These depend on θ_k and p_k. Substituting (2.4) and (2.5) into (2.3) gives

$$\ln L(\boldsymbol{y}) = -\frac{1}{2}[n\ln(\sigma^2) + \ln(|R|)] + \text{const.}$$

This is maximized to find the estimates $\hat{\theta}_k$ and $\hat{p}_k$, which are then substituted into (2.4) and (2.5) to find $\hat{\mu}$ and $\hat{\sigma}$.

The Kriging predictor or best linear unbiased predictor (BLUP) is often derived as the unbiased linear predictor $\boldsymbol{c}_x^t\boldsymbol{y}$ of $Y(\boldsymbol{x})$ that minimizes the mean squared error (MSE) of prediction subject to the unbiasedness constraint. Here we look at Jones' way of deriving the BLUP. Suppose that we have guessed the

function value y^* at a new point $\boldsymbol{x}^*$ and we add this to our sampling points as the $(n+1)$th observation. The likelihood function is updated to include this point and, using the previously calculated parameters, we maximize the log-likelihood as a function of y^*. The idea is to find the y^* that is most consistent with the previously observed variation in the function values at the n sample points. Let

$$\boldsymbol{r}^* = \begin{pmatrix} \mathrm{cor}_K(Y(\boldsymbol{x}^*), Y(\boldsymbol{x}^{(1)})) \\ \vdots \\ \mathrm{cor}_K(Y(\boldsymbol{x}^*), Y(\boldsymbol{x}^{(n)})) \end{pmatrix}.$$

Augmenting the log-likelihood function to include the new point $(\boldsymbol{x}^*, y^*)$ we get

$$\begin{aligned} \ln L(\boldsymbol{y}) &= -\frac{n+1}{2}\ln\left(2\pi\sigma^2\right) - \frac{1}{2}\ln\left(\begin{vmatrix} R & \boldsymbol{r}^* \\ \boldsymbol{r}^{*t} & 1 \end{vmatrix}\right) \\ &\quad -\frac{1}{2\sigma^2}\begin{pmatrix} \boldsymbol{y}-\mathbf{1}\hat{\mu} \\ y^*-\hat{\mu} \end{pmatrix}^t \begin{pmatrix} R & \boldsymbol{r}^* \\ \boldsymbol{r}^{*t} & 1 \end{pmatrix}^{-1} \begin{pmatrix} \boldsymbol{y}-\mathbf{1}\hat{\mu} \\ y^*-\hat{\mu} \end{pmatrix}. \end{aligned}$$

The only part of $\ln L(\boldsymbol{y})$ that depends on y^* is the term

$$-\frac{1}{2\sigma^2}\begin{pmatrix} \boldsymbol{y}-\mathbf{1}\hat{\mu} \\ y^*-\hat{\mu} \end{pmatrix}^t \begin{pmatrix} R & \boldsymbol{r}^* \\ \boldsymbol{r}^{*t} & 1 \end{pmatrix}^{-1} \begin{pmatrix} \boldsymbol{y}-\mathbf{1}\hat{\mu} \\ y^*-\hat{\mu} \end{pmatrix}. \tag{2.6}$$

Substituting

$$\begin{pmatrix} R & \boldsymbol{r}^* \\ \boldsymbol{r}^{*t} & 1 \end{pmatrix}^{-1} = \frac{1}{(1-\boldsymbol{r}^{*t}R^{-1}\boldsymbol{r}^*)} \begin{pmatrix} (1-\boldsymbol{r}^{*t}R^{-1}\boldsymbol{r}^*)R^{-1} + R^{-1}\boldsymbol{r}^*\boldsymbol{r}^{*t}R^{-1} & -R^{-1}\boldsymbol{r}^* \\ -(R^{-1}\boldsymbol{r}^*)^t & 1 \end{pmatrix}$$

into (2.6) gives the function we have to maximize with respect to y^*. The terms involving y^* in this function are

$$\left[\frac{-1}{2\hat{\sigma}^2(1-\boldsymbol{r}^{*t}R^{-1}\boldsymbol{r}^*)}\right](y^*-\hat{\mu})^2 + \left[\frac{\boldsymbol{r}^{*t}R^{-1}(\boldsymbol{y}-\mathbf{1}\hat{\mu})}{\hat{\sigma}^2(1-\boldsymbol{r}^{*t}R^{-1}\boldsymbol{r}^*)}\right](y^*-\hat{\mu}).$$

This is a quadratic function of y^* with negative second derivative,

$$\frac{-1}{\hat{\sigma}^2(1-\boldsymbol{r}^{*t}R^{-1}\boldsymbol{r}^*)} < 0,$$

so maximizing can be achieved by taking the first derivative and setting it equal to zero,

$$\left[\frac{-1}{\hat{\sigma}^2(1-\boldsymbol{r}^{*t}R^{-1}\boldsymbol{r}^*)}\right](y^*-\hat{\mu}) + \left[\frac{\boldsymbol{r}^{*t}R^{-1}(\boldsymbol{y}-\mathbf{1}\hat{\mu})}{\hat{\sigma}^2(1-\boldsymbol{r}^{*t}R^{-1}\boldsymbol{r}^*)}\right] = 0.$$

Solving this for y^* gives the Kriging predictor or BLUP

$$y^* = \hat{y}(\boldsymbol{x}^*) = \hat{\mu} + \boldsymbol{r}^{*t}R^{-1}(\boldsymbol{y}-\mathbf{1}\hat{\mu}).$$

This corresponds to (2.1) with $m=1$, $\pi_1 = 1$, $a_1 = \hat{\mu}$, and $\sum_{j=1}^{n} b_k\phi_k(\boldsymbol{x}-\boldsymbol{x}^{(i)}) = \boldsymbol{r}^{*t}R^{-1}(\boldsymbol{y}-\mathbf{1}\hat{\mu})$.

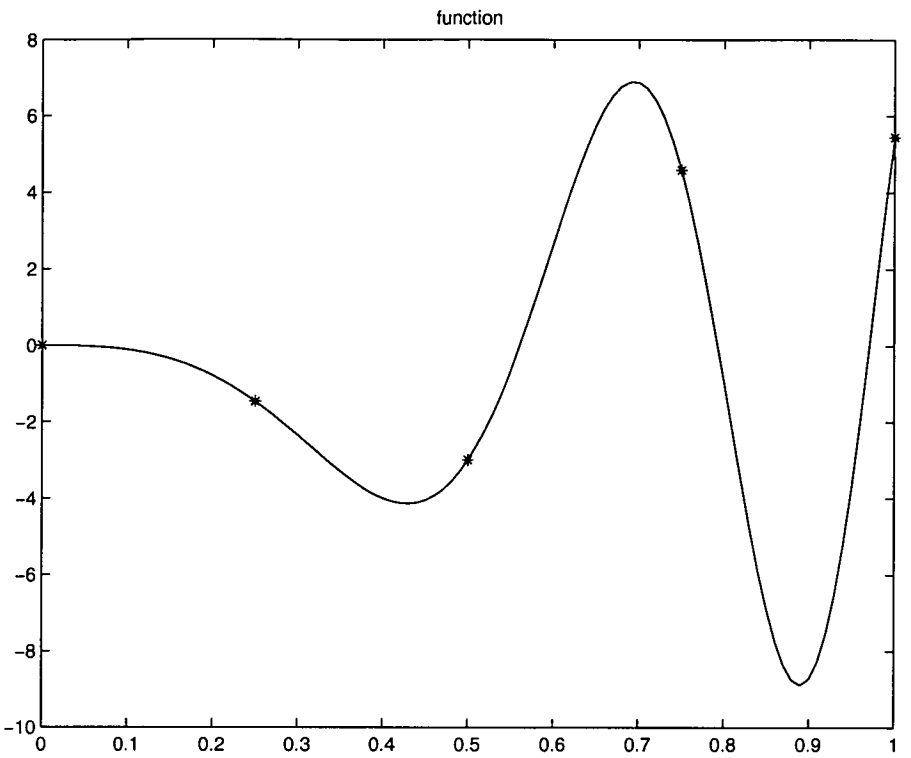

Figure 2.1: Test Function $y(x)$

The mean squared error (MSE) of the predictor is

$$\begin{aligned} s^2(\boldsymbol{x}^*) &= E[y^* - Y(\boldsymbol{x}^*)]^2 \\ &= \hat{\sigma^2}\left[1 - \boldsymbol{r}^{*t}R^{-1}\boldsymbol{r}^* + \frac{(1 - \mathbf{1}R^{-1}\boldsymbol{r}^*)^2}{\mathbf{1}^tR^{-1}\mathbf{1}}\right]. \end{aligned}$$

More details of how to derive the MSE will be given later. To illustrate some characteristics of the BLUP and the MSE and their dependence on $\boldsymbol{\theta}$ and $\boldsymbol{p}$ we look at the function

$$y(x) = -10x\sin(10x^2)$$

where $x \in [0, 1]$, and we choose evenly spaced sampling points $x^{(1)} = 0$, $x^{(2)} = 0.25$, $x^{(3)} = 0.5$, $x^{(4)} = 0.75$ and $x^{(5)} = 1$. Figure 2.1 shows the function $y(x)$, on the interval $[0, 1]$. It has two local minima, one of which is the global minimum.

Figure 2.2 shows the best linear unbiased predictor for $y(x)$ for different values of θ and p. First of all it can be seen that the BLUP always interpolates the function at the points $x^{(1)}, \ldots, x^{(5)}$. For $p = 2$ the BLUP is a smoother function than for $p = 1$. Smaller values of θ correspond to higher correlations between function values at different points. That means that when looking at a new point between two previously sampled points we would expect the function value at the

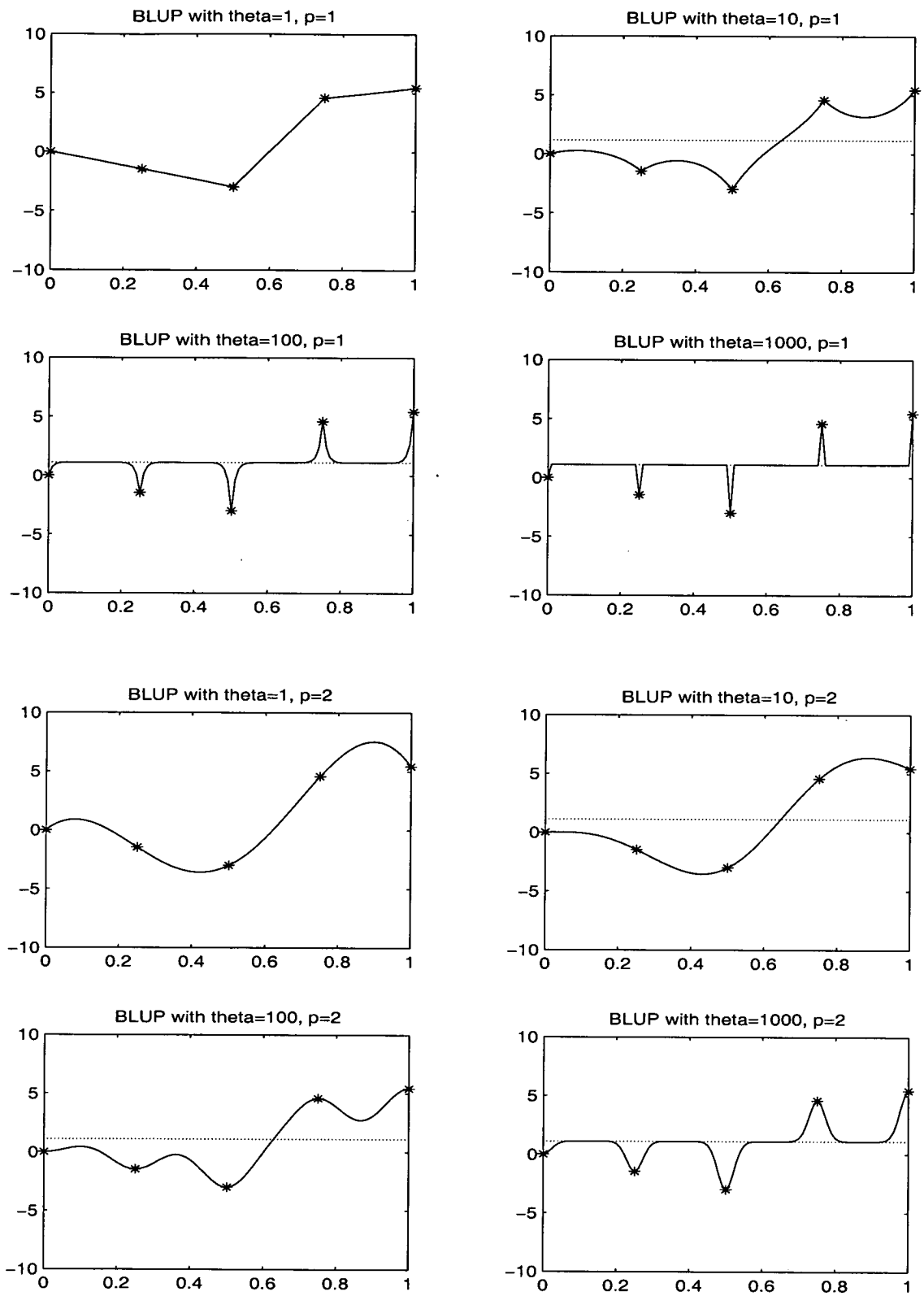

Figure 2.2: BLUP with different θ and p

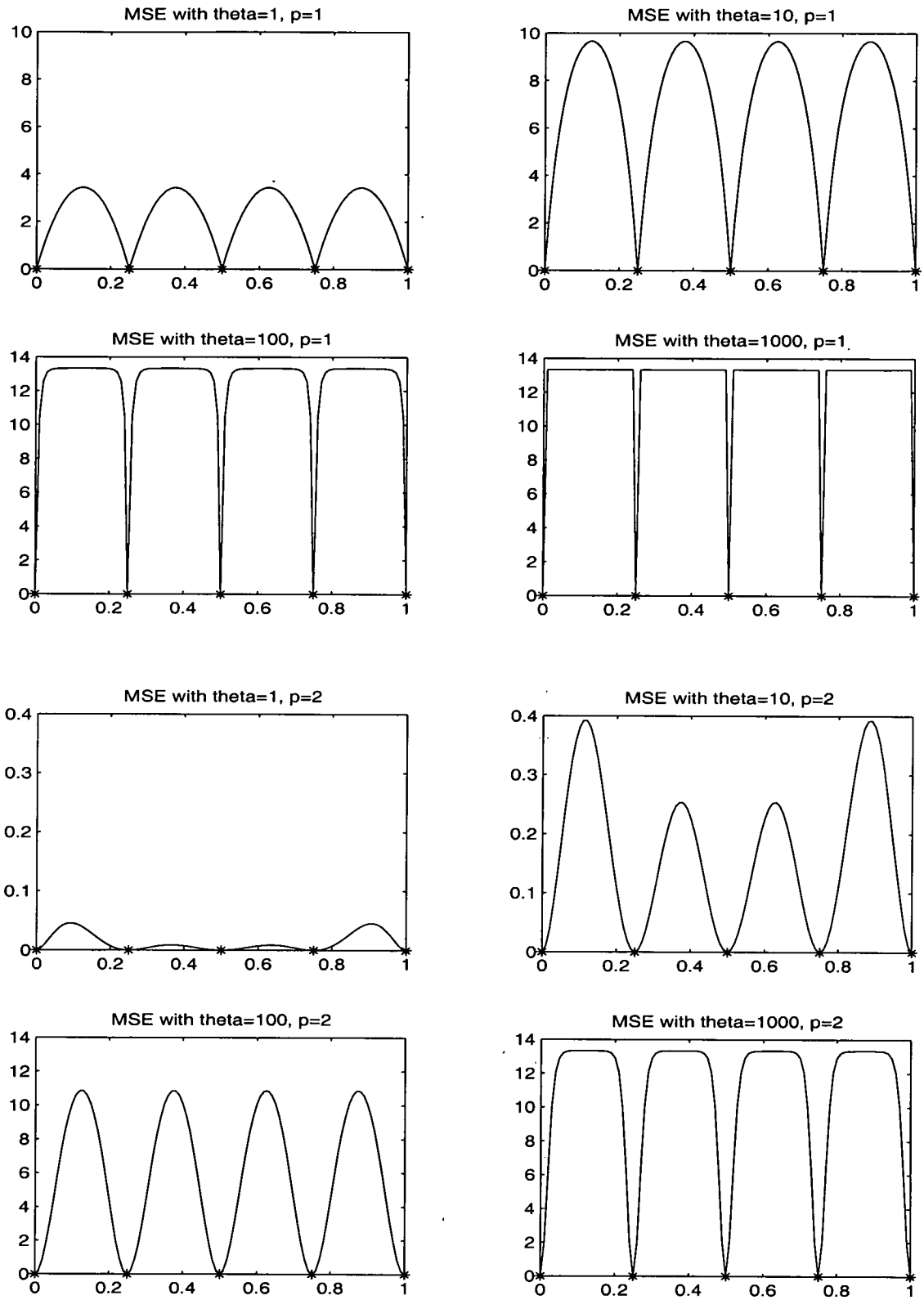

Figure 2.3: MSE with different θ and p

new point to be close to the values at the surrounding sample points, so for points a small distance apart the function values at those points will be close together. The larger θ, the faster the BLUP returns to a certain value, $\hat{\mu}$, between sample points. For this example, minimizing the BLUP as a surrogate function to find a new sample point, updating it, and repeating would lead to a search concentrated around the local minimum of the function, not capturing the global minimum.

Figure 2.3 shows the mean squared error of the predictor for different values of θ and p. It can be seen that the MSE is always zero at sample points and rises in between them, more rapidly and to a greater value the larger the value of θ.

2.2.3 Alternative Random Function Methods

In this section we will summarize different random function methods. It might seem appealing just to minimize $\hat{y}(\boldsymbol{x})$ but this has the disadvantage that the global optimum of the function might be missed completely as the algorithm would concentrate new sample points exclusively around the area where $\hat{y}(\boldsymbol{x})$ is lowest and not explore any other regions. This is what would have happened for the test function in Figure 2.1 with 5 initial sample points. This approach might be suitable for local optimization but not for global optimization. A way of avoiding this is to take the error of the predictor into account when choosing new sample points. The idea is that when the area around the currently known minimum of $\hat{y}(\boldsymbol{x})$ has been sufficiently explored, new points where the error of the predictor is high will be added to the sample, and because of this it is more likely that the global optimum of the function will be found. Different approaches to this are described in the following.

2.2.3.1 Merit Functions

Torczon and Trosset [70] investigate the approach of minimizing the predictor $\hat{y}(\boldsymbol{x})$ using grid search strategies. The grid is used partly for reasons of robustness and convergence results, using additional refinements of the grid. Again, if there is no initial data in the basin of attraction of the global minimizer, this area is not investigated further: a local optimum is found and the global optimum missed. To overcome this problem they suggest an alternative strategy which uses a merit function of the form

$$M_1(\boldsymbol{x}) = \hat{y}(\boldsymbol{x}) - w_1\delta_1(\boldsymbol{x}) \tag{2.7}$$

where $w_1 \geq 0$, $\hat{y}$ is an interpolating function and

$$\delta_1(\boldsymbol{x}) = \min \|\boldsymbol{x} - \boldsymbol{x}^{(i)}\|_2, \quad i = 1, \ldots, n.$$

The merit function M_1 consists of the two components $\hat{y}$ and $\delta_1(\boldsymbol{x})$ so, to minimize M_1, the minimization of the interpolating function $\hat{y}(\boldsymbol{x})$ is balanced with a maximization of $\delta_1(\boldsymbol{x}) = \min \|\boldsymbol{x} - \boldsymbol{x}^{(i)}\|_2$, the distance from $\boldsymbol{x}$ to the nearest previous sample point. With $w_1 = 0$ the search is completely based on $\hat{y}(\boldsymbol{x})$ and most likely will find a local optimum of the function. As w_1 is chosen to be a larger number, more emphasis is placed on maximizing $\delta_1(\boldsymbol{x}) = \min \|\boldsymbol{x} - \boldsymbol{x}^{(i)}\|_2$ and the search becomes more global. The quantity w_1 could be kept constant, or could be varied in each iteration: letting $w_1 \to 0$ as the optimization progresses, encourages convergence to a local minimum. The choice of w_1 seems to be based on experiments. According to Torczon and Trosset, applying this strategy to a multimodal test function, even where none of the points of the initial sample lies in the basin of attraction of the global minimizer, leads to a favourable result.

Pattern search methods using response surfaces are also further investigated by Torczon and Trosset in [69] with a different pattern search strategy, not described in more detail here. The search criterion suggested is

$$M_2(\boldsymbol{x}) = \hat{y}(\boldsymbol{x}) - w_2\sqrt{s^2(\boldsymbol{x})} \tag{2.8}$$

where $s^2(\boldsymbol{x})$ is the mean squared error of prediction at $\boldsymbol{x}$. Figure 2.4 illustrates the dependence of $M_2(\boldsymbol{x})$ on the choice of w_2. As above, with $w_2 = 0$ the search is completely based on $\hat{y}(\boldsymbol{x})$ and most likely will find a local optimum of the function. As w_2 is chosen to be a larger number, more emphasis is placed on maximizing $s(\boldsymbol{x})$ and the search becomes more global. Again, letting $w_2 \to 0$ as the optimization progresses encourages convergence to a local minimum.

2.2.3.2 Statistical Lower Bound

Cox and John in [13] use the following method. A grid of points is selected in the domain of interest of the function. At a number n of these points, $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$ the function is evaluated and these points form the *observation sites*. The lowest function value, denoted by y_{0n}, is recorded. The remaining k grid points will form the *prediction sites*. Based on the initial design, i.e. the set of observation points, the BLUP $\hat{y}$ and MSE s^2 are computed. These are used to compute a "lower confidence bound" (lcb) at any prediction site by

$$M_3(\boldsymbol{x}^{(i)}) = \hat{y}(\boldsymbol{x}^{(i)}) - w_3\hat{\sigma}\sqrt{s^2(\boldsymbol{x}^{(i)})},\ i = 1, \ldots, k, \tag{2.9}$$

where $\hat{y}(\boldsymbol{x}^{(i)})$ is the BLUP and $\hat{\sigma}$ is the maximum likelihood estimate of the standard deviation. The prediction site with the smallest lower confidence bound is selected as a possible approximation to the minimum of the objective function.

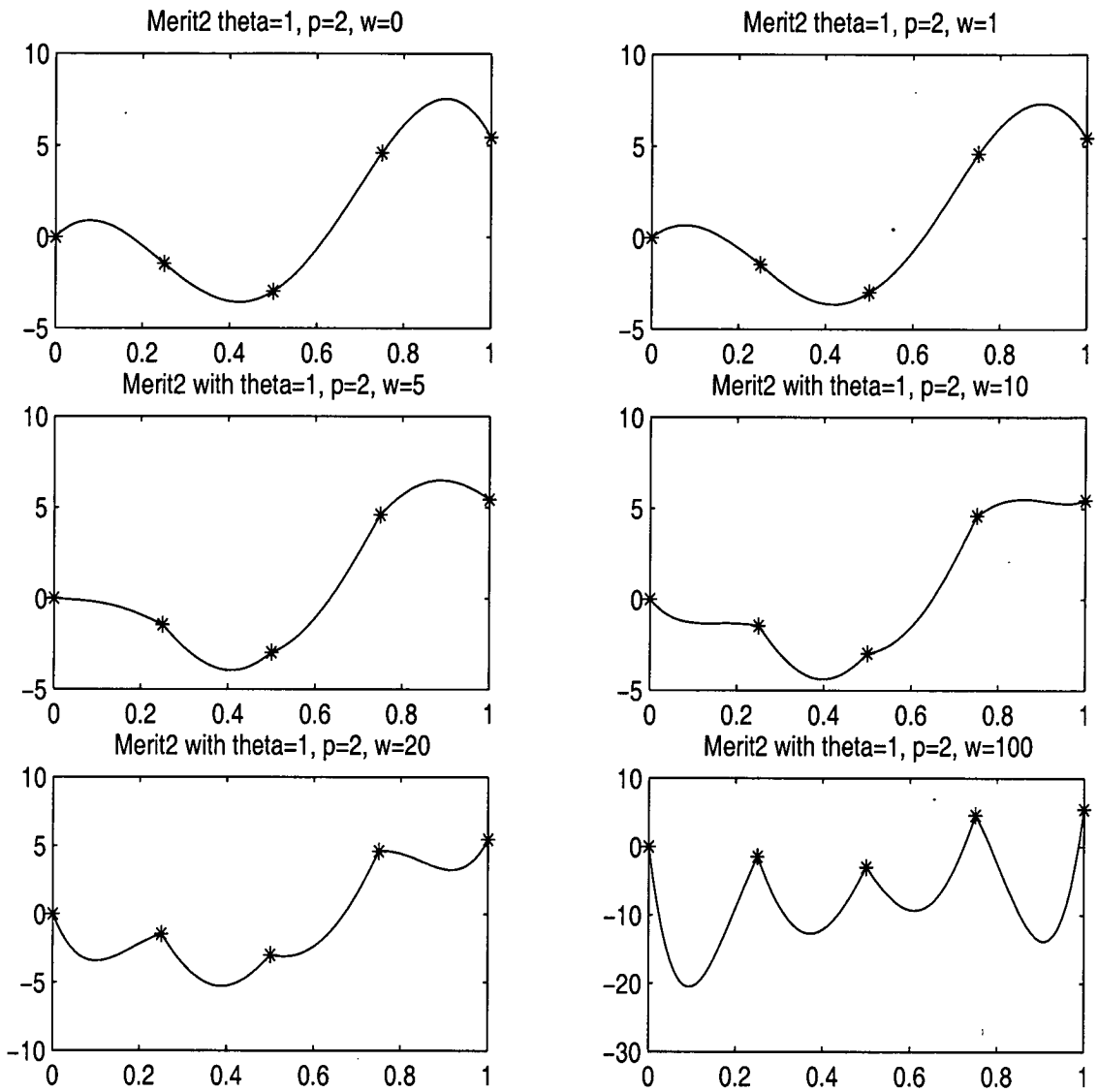

Figure 2.4: Merit Function $M_2(\boldsymbol{x})$

The objective function is evaluated at this point, the point added to the set of observation sites, and y_{0n}, BLUP, MSE, etc. are updated. The stopping criterion is based on either a user-specified maximum number of function evaluations or iterations, or the following convergence criterion,

$$y_{0n} < \min_{i=1,\ldots,k}(M_3(\boldsymbol{x}^{(i)})).$$

As Jones demonstrates in [32] this can fail to find the global optimum for a one-dimensional test-function. In this example the global optimum is not picked up in the initial sample and as $M_3(\boldsymbol{x}^{(i)})$ is minimized the algorithm concentrates on a local optimum.
The lower confidence bound $M_3(\boldsymbol{x}^{(i)})$ in (2.9) is similar to the second merit function $M_2(\boldsymbol{x})$ in (2.8) and the location of the new sampling point $\boldsymbol{x}^{(n+1)}$ depends very much on the choice of w_3 and on $\hat{\sigma}$.

2.2.3.3 Probability of Improvement

Another approach finds new sample points where the probability of improving the function beyond a target value T is high. The target value T should be set such that $T < y_{0n}$ where y_{0n} is the best function value found up to this point. The probability of improvement beyond the target value at point $\boldsymbol{x}$ is then $P(Y(\boldsymbol{x}) < T)$. An approach like this was suggested by Kushner as early as 1964, see the article by Kushner [41]. This approach was later taken up by Žilinskas as the so-called *P-algorithm*, see for example [74] and [75]. A variant of this, the so-called *P*-algorithm* was later proposed: in areas with a great density of sample points the objective function is modeled by a polynomial rather than a statistical model, for more details see for example [71].

We will now briefly look at the idea behind the *probability of improvement* and then give some more details about the *P-algorithm* and Žilinskas' axiomatic justification for it. Assuming that the random variable $Y(\boldsymbol{x})$ is normally distributed with mean $\hat{y}(\boldsymbol{x})$ and standard deviation $s(\boldsymbol{x})$ the probability of improving beyond the target value T is

$$P(Y(\boldsymbol{x}) < T) = \Phi\left(\frac{T - \hat{y}(\boldsymbol{x})}{s(\boldsymbol{x})}\right)$$

where Φ is the standard normal cumulative distribution function. Probability of improvement is high where $\hat{y}(\boldsymbol{x})$ is small or $s(\boldsymbol{x})$ is large. This leads to sampling around the current best point until the standard error $s(\boldsymbol{x})$ in that region becomes too small. Usually $T < \hat{y}(\boldsymbol{x})$, and for small $s(\boldsymbol{x})$, the term $\dfrac{T - \hat{y}(x)}{s(x)}$ becomes extremely negative. Then the probability of improvement becomes small around

the current best point and this leads to sampling elsewhere, where the standard error is higher. A difficulty is the choice of T. Lower values of T will lead to a more local search in the region where $\hat{y}$ is least, higher values of T will lead to more emphasis on global search. In [32] Jones suggests overcoming the difficulty of choosing a value for T by using expected improvement, as will be explained later, or using several values of T in every iteration, leading to a selection of several sample points at once. Jones also points out a correspondence between T in the probability of improvement and w_3 in the lower confidence bound approach.

2.2.3.4 The P-algorithm

In [75] Žilinskas introduces an axiomatic way of treating the probability of improvement method. This is explained here. The Gaussian random variables $Y(\boldsymbol{x})$, $\boldsymbol{x} \in D$ are used as a statistical model of the function $y(\boldsymbol{x})$, where the distribution function $P_x(.)$ of $Y(\boldsymbol{x})$ depends on the sampled points $\boldsymbol{x}^{(i)}$ and their function values y_i, $i = 1, \ldots, n$. Let y^* be a target value that it is desirable to reach, for example an estimate of the global minimum. The choice of the next sample point $\boldsymbol{x}^{(n+1)} \in D$ can be interpreted as a choice of distribution function $P_{x^{(n+1)}} \in \{P_x | \boldsymbol{x} \in D\}$. Under some "rational requirements" the preferences of choice of $P_x(.)$ are reflected by a unique utility function $ut(.)$ satisfying certain axioms. The statement "P_{z_1} is preferable over P_{z_2}" is also written as $P_{z_1} \geq P_{z_2}$ and defined by

$$P_{z_1} \geq P_{z_2} \Leftrightarrow \int_{-\infty}^{\infty} ut(y) P_{z_1}(y) dy \geq \int_{-\infty}^{\infty} ut(y) P_{z_2}(y) dy.$$

Since $P_{z_k}(.)$ are Gaussian, preferences in the choice of $P_{z_k}(.)$ correspond to preferences in the choice of μ_k and s_k, the mean and standard deviation of $Y(\boldsymbol{z}_k)$ respectively, and we write $(\mu_1, s_1) > (\mu_2, s_2)$ if P_{z_1} is preferable over P_{z_2}. In [75] Žilinskas gives four axioms to "characterize the requirements of rational search",

A.1 If $\mu_1 < \mu_2$, $s_1 > 0$, $\mu_2 > y^*$, then there exists $s > 0$ such that $(\mu_1, s_1) > (\mu_2, s_2)$ if $s_2 < s$.

A.2 If μ_1, $s_1 > 0$, $\mu_2 > y^*$, then it is true that $(\mu_1, s_1) > (\mu_2, 0)$.

A.3 If $s_1 > 0$, $\mu_2 > y^*$, s_2, then there exists μ_1 with $\mu_2 > \mu_1 > y^*$, such that $(\mu_1, s_1) > (\mu_2, s_2)$.

A.4 $ut(.)$ is continuous from the left.

In [75] Žilinskas then proves a theorem stating the following.

Theorem 2.2.1

The unique (to within linear transformation) function satisfying A.1-A.4 is

$$ut(t) = \begin{cases} 1 & t \leq y^* \\ 0 & t > y^*. \end{cases}$$

Therefore the utility of evaluating the function at the point $\boldsymbol{x}$ is proportional to $P_x(y^*)$ where y^* is a value we are hoping to reach, for example an estimate of the global minimum. This is the probability that $Y(\boldsymbol{x}) < y^*$,

$$P(Y(\boldsymbol{x}) < y^*) = \Phi\left(\frac{y^* - \hat{y}(\boldsymbol{x})}{s(\boldsymbol{x})}\right)$$

where Φ is the standard normal cumulative distribution function. The new point $\boldsymbol{x}^{(n+1)}$ will be chosen to maximize the utility,

$$\boldsymbol{x}^{(n+1)} = \arg\max_{x \in D} P_x(y^*). \tag{2.10}$$

Žilinskas investigates the dependence of the point $\boldsymbol{x}^{(n+1)}$ on y^*. In particular he proves a theorem that says that if $y^* = \min_{x \in D} \hat{y}(\boldsymbol{x})$ then the set of points that maximize $P_x(y^*)$ is identical to the set of points that minimize $\hat{y}(\boldsymbol{x})$. If $y^* \to -\infty$ then $\inf_{z \in B} \|\boldsymbol{x}^{(n+1)} - \boldsymbol{z}\| \to 0$ where B is the set of points that maximize $s(\boldsymbol{x})$. He proceeds to show that as y^* decreases, $s(\boldsymbol{x})|_{x=x^{(n+1)}}$ increases. The conclusion drawn from these two theorems is that the P-algorithm searches more globally as y^* decreases. Different y^* can be used to achieve better efficiency of the P-algorithm. Žilinskas also provides a proof of convergence of the P-algorithm under certain assumptions. See also [76] for more details and [74] for further background reading.

2.2.3.5 Expected Improvement

A method can be devised that balances global and local search without requiring the setting of a target value. This uses *expected improvement* (EI) as sampling criterion, and this is the method which will be used in this thesis.

Let y_{0n} as before be the best function value found at sample points and let $Y(\boldsymbol{x})$ be a random variable describing the uncertainty about $y(\boldsymbol{x})$. Then the expected improvement is given by

$$\begin{aligned} E[I(\boldsymbol{x})] &= E[\max(y_{0n} - Y, 0)] \\ &= \int_{-\infty}^{y_{0n}} (y_{0n} - y)\phi_x(y)dy \end{aligned}$$

where $\phi_x(.)$ is the probability density function of $Y(\boldsymbol{x})$. Splitting up the integral into two terms and integrating by substitution gives

$$E[I(\boldsymbol{x})] = (y_{0n} - \hat{y})\Phi\left(\frac{y_{0n} - \hat{y}}{s}\right) + s\phi\left(\frac{y_{0n} - \hat{y}}{s}\right)$$

where $\phi(.)$ and $\Phi(.)$ are the standard normal probability density and the cumulative distribution function respectively. Schonlau for example uses this approach in [63].

Where the probability of improvement method uses the utility function

$$ut(y) = \begin{cases} 1 & y \leq y^* \\ 0 & y > y^* \end{cases}$$

the expected improvement takes into account the amount of improvement by using this amount of improvement as the utility function

$$I(y) = \begin{cases} y_{0n} - y & y \leq y_{0n} \\ 0 & y > y_{0n}. \end{cases}$$

The preference of choice of distribution function is

$$P_{z_1} \geq P_{z_2} \Leftrightarrow \int_{-\infty}^{\infty} I(t) P_{z_1}(t) dt \geq \int_{-\infty}^{\infty} I(t) P_{z_2}(t) dt$$

and maximizing the expected utility is the same as maximizing the expected improvement. The utility function $I(.)$ satisfies Žilinskas axioms A.1, A.2, and A.4. In the case where $\mu_1 = \mu_2$ it tells us to choose the distribution function $P_{z_i}(.)$ for which s_i, $i = 1, 2$ is greater.

The use of expected improvement was also studied for example by Locatelli in [42], where a Wiener process is used to model the objective function.

2.2.3.6 Goal Seeking

Assume that we want to achieve a target value or "goal" function value, y^*. Making a hypothesis about the location where this value is achieved, for example at the point x^* we can compute the conditional likelihood of the data given that the surface passes through the point (x^*, y^*). The conditional likelihood is given by

$$\frac{1}{(2\pi\sigma^2)^{n/2}|C|^{1/2}} \exp\left[\frac{-(\boldsymbol{y} - \boldsymbol{\mu})^t C^{-1} (\boldsymbol{y} - \boldsymbol{\mu})}{2\sigma^2}\right]$$

where $\boldsymbol{\mu} = \mathbf{1}\mu + \boldsymbol{r}(f^* - \mu)$ and $C = R - \boldsymbol{r}\boldsymbol{r}^t$ are the conditional mean and correlation matrix respectively. The conditional log-likelihood can be used to evaluate the "credibility" that the surface passes through $(\boldsymbol{x}^*, y^*)$. The parameters μ, σ^2, θ_k, and p_k are adjusted to maximize the conditional likelihood. This method has been suggested by Jones, see [32].

2.2.3.7 Minimizing Bumpiness

If y^* is a guess of the global minimum we can choose any point $\boldsymbol{x}$ in the domain and assume that this is where the minimum y^* is attained. Then a response surface is fitted through the point $(\boldsymbol{x}, y^*)$ and the previously sampled points. If this response surface is very bumpy, intuitively we would not expect $\boldsymbol{x}$ to be the global minimizer. As a new sample point we choose the point that achieves the target and minimizes the bumpiness. Gutmann in [28] and Björkman and Holmström in [4] investigate this approach based on a general response surface technique proposed by Jones [30], using radial basis functions as interpolants. Let $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$ be the points at which the objective function values y_i are known. The radial basis function interpolant is of the form

$$\hat{y}(\boldsymbol{x}) = \sum_{k=1}^{m} a_k \pi_k(\boldsymbol{x}) + \sum_{j=1}^{n} b_j \rho \|\boldsymbol{x} - \boldsymbol{x}^{(j)}\| \tag{2.11}$$

where $\{\pi_k(\boldsymbol{x}) | k = 1, \ldots, m\}$ is a basis of the space of all polynomials in $\boldsymbol{x}$ of degree g. The coefficients a_k and b_j are defined by the system

$$\hat{y}(\boldsymbol{x}^{(i)}) = y_i, \quad i = 1, \ldots, n \tag{2.12}$$

$$\sum_{k=1}^{n} b_k \pi_j(\boldsymbol{x}^{(k)}) = 0, \quad j = 1, \ldots, m. \tag{2.13}$$

The equations (2.12) guarantee that the interpolant does interpolate the points $(\boldsymbol{x}^{(1)}, y_1), \ldots, (\boldsymbol{x}^{(n)}, y_n)$; the additional m equations (2.13) are there to "guarantee existence and uniqueness of an interpolant" (Gutmann, [28]). The measure of bumpiness used by Gutmann is derived from the theory of natural cubic splines in one dimension: for $\hat{y}$ which satisfies (2.12) and (2.13) it is known that

$$\begin{aligned} \hat{y} = \arg \quad \min \quad & I(g) := \int_{\mathbb{R}} (g''(x))^2 dx \\ \text{s.t.} \quad & g(x_i) = y_i, \ i = 1, \ldots, n \end{aligned}$$

among all $g : \mathbb{R} \to \mathbb{R}$ for which $I(g)$ exists and is finite. With the basis function matrix Φ defined by $(\Phi)_{ij} := \rho(\|x_i - x_j\|) = (\|x_i - x_j\|)^3$, $i, j = 1, \ldots, n$, one gets $I(\hat{y}) = 12\boldsymbol{b}^T \Phi \boldsymbol{b}$. The measure of bumpiness Gutmann uses in [28] is based on the fact that

$$(-1)^{m_0+1} \boldsymbol{b}^T \Phi \boldsymbol{b} > 0$$

for all $\boldsymbol{b} \neq \boldsymbol{0}$ which satisfy condition (2.13) and where the choice of $m_0 = -1, 0, 1$ depends on the radial basis functions used. This property can be used to define

a semi-inner product and a semi-norm for ρ and $m \geq m_0$. For

$$v(\boldsymbol{z}) = \sum_{k=1}^{m} a_k \pi_k(\boldsymbol{z}) + \sum_{j=1}^{N(v)} b_j \rho \|\boldsymbol{z} - \boldsymbol{x}^{(j)}\|$$

and

$$w(\boldsymbol{z}) = \sum_{k=1}^{m} g_k \pi_k(\boldsymbol{z}) + \sum_{j=1}^{N(w)} h_j \rho \|\boldsymbol{z} - \boldsymbol{y}^{(j)}\|,$$

for $N(v), N(w) \in \mathbb{N}$, and where $\boldsymbol{b}$ and $\boldsymbol{h}$ satisfy (2.13), the semi-inner product is defined by

$$\langle v, w \rangle := (-1)^{m_0+1} \sum_{i=1}^{N(v)} b_i w(\boldsymbol{x}^{(i)}).$$

The semi-norm is used as a measure of bumpiness $\sigma(\hat{y})$ for interpolants $\hat{y}$ of the form (2.11) by means of $\sigma(\hat{y}) = \langle \hat{y}, \hat{y} \rangle$.

Now consider the following interpolation conditions

$$\hat{y}(\boldsymbol{x}^{(i)}) = y_i, \quad i = 1, \ldots, n \tag{2.14}$$

$$\hat{y}(\boldsymbol{x}) = y^* \tag{2.15}$$

and let $\hat{y}_n$ be a radial basis function satisfying (2.14), and for each unsampled $\boldsymbol{x} \notin \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}\}$ let $\hat{y}_x$ be a radial basis function satisfying (2.14) and (2.15). For a target value y^* and a point $\boldsymbol{x} \in D \setminus \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}\}$ the radial basis function $\hat{y}_x$ can be written as

$$\hat{y}_x(\boldsymbol{z}) = \hat{y}_n(\boldsymbol{z}) + (y^* - \hat{y}_n(\boldsymbol{x})) l_n(\boldsymbol{z}, \boldsymbol{x}), \quad \boldsymbol{z} \in \mathbb{R}^d$$

where $l_n(\boldsymbol{z}, \boldsymbol{x})$ is the radial basis function which satisfies the conditions

$$l_n(\boldsymbol{x}, \boldsymbol{x}^{(i)}) = 0, \quad i = 1, \ldots n$$

$$l_n(\boldsymbol{x}, \boldsymbol{x}) = 1.$$

Then $l_n(\boldsymbol{z}, \boldsymbol{x})$ can be written in the form (2.11) as follows

$$l_n(\boldsymbol{z}, \boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i(\boldsymbol{z}) \rho(||\boldsymbol{x} - \boldsymbol{x}^{(i)}||) + \mu_n(\boldsymbol{z}) \rho(||\boldsymbol{x} - \boldsymbol{z}||) + \sum_{i=1}^{m} \beta_i(\boldsymbol{z}) \pi_i(\boldsymbol{z})$$

where the coefficients $(\alpha_1(\boldsymbol{z}), \ldots, \alpha_n(\boldsymbol{z}))$, $(\beta_1(\boldsymbol{z}), \ldots, \beta_m(\boldsymbol{z}))$, and $\mu_n(\boldsymbol{z})$ are defined by a system of equations. Gutmann uses the semi-inner product and semi-norm as defined above and takes $\sigma(\hat{y}) = \langle \hat{y}, \hat{y} \rangle$ as the measure of bumpiness of $\hat{y}$, and deduces the following formula

$$\sigma(\hat{y}_x) = \sigma(\hat{y}_n) + (-1)^{m_0+1} \mu_n(\boldsymbol{x}) [y^* - \hat{y}_n(\boldsymbol{x})]^2$$

for the bumpiness $\sigma(\hat{y}_x)$ of $\hat{y}_x$.

A rough outline of Gutmann's algorithm is as follows,

- Pick a radial basis function ρ. Find the radial basis function interpolant $\hat{y}_n$ of the form (2.11) that minimizes the bumpiness $\sigma(\hat{y})$ subject to the interpolation condition (2.14).

- Choose a target value $y^* \in [-\infty, \min_{x \in D} \hat{y}(\boldsymbol{x})]$. Find $\boldsymbol{x}^{(n+1)}$ that minimizes the bumpiness $\sigma(\hat{y}_x) = \sigma(\hat{y}_n) + (-1)^{m_0+1}\mu_n(\boldsymbol{x})[y^* - \hat{y}_n(\boldsymbol{x})]^2$. Set $n = n + 1$. Stop if $n > n_{\max}$, a prescribed maximum number of function evaluations.

Further, Gutmann investigates the similarities between the radial basis function method he presents in [28] and the P-algorithm. He also proves a convergence result for the radial basis function method in the case where the basis functions used are linear, cubic, or thin plate splines.

2.2.4 Other Developments

The work presented in the remainder of this thesis is mainly based on the expected improvement criterion as presented in the papers by Sacks *et al.* [60, 61] or Schonlau's thesis [63]. Since that time, research in the area has included work on constrained optimization problems, see for example Schonlau *et al.* in [66], and Sasena's thesis [62]. Some investigation of better means of controlling global versus local search has been pursued further by Schonlau *et al.* in [66]. These approaches are all based on the expected improvement criterion. Schonlau in [63] also addresses the issue of sampling several points at a time.

2.2.5 Conclusions

This chapter summarizes different techniques for global optimization. We concentrate on the approaches where a sampling criterion is optimized to find promising new points at which to evaluate the objective function. Both minimizing a quadratic surface that is fitted to the sampling points and minimizing a surrogate function that interpolates the sampling points might lead to local search only and the global minimum might be missed. Overall the surrogate function can be improved if new points are added where the uncertainty about the surrogate is high. This leads to more emphasis on global search. Some of the sampling criteria, for example the merit functions and the statistical lower bound, are direct combinations of minimizing the expected objective function value minus a multiple of the error. The different terms, and therefore the emphasis on local or global search, are balanced by weights, the choice of which is not straightforward, and is often based on experiments. The probability of improvement and P-algorithm depend very much on the choice of the target value, as can be seen from Žilinskas' theorems. Jones in [32] suggests overcoming this difficulty of choosing a target value

by using several target values in every iteration or by using expected improvement which does not require this choice to be made. Good results are reported with Gutmann's approach.

Chapter 3

Background

This chapter gives the necessary background for the methods used in the rest of the thesis. We begin by summarizing the general approach to global optimizing of expensive functions.

Computer simulations are widely used in engineering decision making. Often these pose optimization problems where the evaluation of the objective function is very expensive. In solving the optimization problem the aim is to keep computational cost low, but if one function evaluation on its own is very expensive this justifies putting some effort into trying to avoid having to make function evaluations. The strategy is the following: the function is evaluated at a set of initial points. Then a comparatively simple approximation to the function is found and used as a surrogate function. Based on the idea that interpolating methods become more and more accurate as new points are added, often a surrogate function which interpolates the original objective function at the known points is chosen. Using the surrogate function we then try to find new promising points at which to evaluate the true function. An issue here is the balance between local and global search. We want to find points at which we expect the function to be lower than the lowest function value we have found so far. However, at the same time we want to keep improving our surrogate function, keeping in mind that it might not always be very accurate, so we look for new sample points that are far away from the points sampled at up to this point. That means we look for points where the surrogate function value is low and where the error in the current surrogate is expected to be high. Since these two objectives usually conflict, we need to decide on their relative importance, and this is usually done by defining a merit function which combines the surrogate and the error function. A general algorithmic structure is:

1. evaluate the objective function at a set of n initial points $\{\boldsymbol{x}^{(i)}\}_{i=1}^{n}$,

2. form the approximating function and the error function,

3. optimize a merit function based on the approximating function and the error function to find a promising new sample point $\boldsymbol{x}^{(n+1)}$,

4. if the stopping criterion is met, stop, otherwise continue,

5. evaluate the objective function at $\boldsymbol{x}^{(n+1)}$,

6. set $n := n + 1$, go to step 2.

After each new evaluation of the objective function, the approximating function and the error function, and thus the merit function, are recalculated to take the new data point into account. The stopping criterion can be based on the value of the merit function. One example of a merit function is the expected improvement function. The expected improvement at any point in the range reflects the expected amount of improvement of the approximating function beyond a target value (usually the best function value found up to this point) at that point, taking into account the uncertainty in the approximating function, i.e. the error function. The next sample point is chosen where the expected improvement is maximized.

In this approach to global optimization of expensive functions it is assumed that the objective function can be modelled as a regression term plus a stochastic process term. Therefore stochastic processes play an important role, and we will start here by introducing some terminology and background on stochastic processes before explaining the background of the global optimization method of interest for this thesis.

3.1 Stochastic Processes

We are interested in certain properties of stochastic processes, such as covariances and correlations. The aim of this section is to give the background we need to work with stochastic processes, and their specific properties we are interested in. More details can be found for example in Grimmett and Stirzaker's book [27], along with the following definitions.

Definition 3.1.1
(i) A random or stochastic process Y is a family $\{Y(\boldsymbol{x}) : \boldsymbol{x} \in X\}$ of random variables which map the sample space Ω into some set S.
(ii) For any fixed $\omega \in \Omega$ there is a corresponding set $\{Y_\omega(\boldsymbol{x}) : \boldsymbol{x} \in X\} \subset S$; this is called the realization or sample path of Y at ω.

We assume throughout that $X = \mathbb{R}^d$ and that $S = \mathbb{R}$, i.e. the processes are real-valued. For fixed $\omega \in \Omega$, $Y_\omega(\boldsymbol{x})$ is a function value at point $\boldsymbol{x}$. The stochastic

process can be interpreted as defining a family of functions $Y_\omega(\boldsymbol{x})$ of $\boldsymbol{x} \in X$, each function with some probability assigned to it. How do two random variables $Y(\boldsymbol{x})$ and $Y(\boldsymbol{v})$ depend on each other? This dependency can be captured in the covariance $\mathrm{cov}(Y(\boldsymbol{x}), Y(\boldsymbol{v}))$ and the correlation $\mathrm{cor}(Y(\boldsymbol{x}), Y(\boldsymbol{v}))$, $\boldsymbol{v}, \boldsymbol{x} \in X$. Note again that $Y(\boldsymbol{x}), Y(\boldsymbol{v})$ are random variables and for fixed $\omega \in \Omega$, $Y_\omega(\boldsymbol{x}), Y_\omega(\boldsymbol{v})$ are possible outcomes. We will not focus on ω and in line with our interpretation of $Y_\omega(\boldsymbol{x})$ as a function value at the point $\boldsymbol{x}$, we will also use the notation $Y(\boldsymbol{x})$ instead of $Y_\omega(\boldsymbol{x})$.

We will now introduce some essential definitions for working with random variables and then progress to looking at stochastic processes, i.e. families of random variables.

Definition 3.1.2

Let Y, Y_1, and Y_2 be scalar random variables and let y be the possible values of Y.

(i) If Y has probability density function p, then the expectation of Y is given by

$$E(Y) = \int_{-\infty}^{\infty} yp(y)dy.$$

(ii) The covariance of Y_1 and Y_2 is defined to be

$$\mathrm{cov}(Y_1, Y_2) = E[(Y_1 - E(Y_1))(Y_2 - E(Y_2))].$$

(iii) The variance of Y is

$$\mathrm{var}(Y) = \mathrm{cov}(Y, Y).$$

(iv) The correlation (coefficient) of Y_1 and Y_2 is

$$\mathrm{cor}(Y_1, Y_2) = \frac{\mathrm{cov}(Y_1, Y_2)}{(\mathrm{var}(Y_1)\mathrm{var}(Y_2))^{1/2}}.$$

Note that, from the definition of covariance and variance of a random variable Y, the variance of Y can be calculated by the following

$$\begin{aligned} \mathrm{var}(Y) &= E[(Y - E(Y))(Y - E(Y))] \\ &= E(Y^2) - (E(Y))^2. \end{aligned}$$

For every $\omega \in \Omega$ we can form a vector of random variables $Y_x = (Y(\boldsymbol{x}^{(1)}), \ldots, Y(\boldsymbol{x}^{(n)}))^t$ at the points $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)} \in X$. If $S \subseteq \mathbb{R}$, then the vector $(Y(\boldsymbol{x}^{(1)}), \ldots, Y(\boldsymbol{x}^{(n)}))^t$ has joint distribution function $P_x : \mathbb{R}^n \to [0, 1]$ given by

$$\begin{aligned} P_x(\boldsymbol{z}) &= P(Y(\boldsymbol{x}^{(1)}) \leq z_1, \ldots, Y(\boldsymbol{x}^{(n)}) \leq z_n) \\ &= \int_{-\infty}^{z_1} \cdots \int_{-\infty}^{z_n} p(\boldsymbol{t})dt_n \ldots dt_1, \ \boldsymbol{x} \in \mathbb{R}^n \end{aligned}$$

for suitable $p : \mathbb{R} \to [0, \infty)$; p is the joint probability density function of Y_x.

Definition 3.1.3

(i) If Z is an $m \times n$ matrix of random variables, then $E(Z)$ is the $m \times n$ matrix with (i,j)th entry $E(Z_{ij})$.

(ii) For a random vector Z the matrix $E[(Z - E(Z))(Z - E(Z))^t] = \mathrm{Cov(Z)}$ is the covariance matrix of Z. Its (i,j)th entry is $\mathrm{Cov}_{ij} = \mathrm{cov}(Z_i, Z_j)$.

The following lemma summarizes some useful properties of the expectation and the variance, for more details see for example [2].

Lemma 3.1.1

(i) If Z is an $m \times n$ matrix of random variables, A is an $l \times m$ real matrix and F is an $n \times q$ real matrix, then

$$E(AZF) = A(E(Z))F.$$

(ii) If $V = AY$, where Y is a random vector, then

$$\begin{aligned} E(V) &= AE(Y) \\ \mathrm{Cov}(V) &= A\mathrm{Cov}(Y)A^t. \end{aligned}$$

Part (i) can be proved by comparing elements of the matrices $E(AZF)$ and $A(E(Z))F$, part (ii) follows from (i).

Closely related to the notions of covariance and correlation are the autocovariance and autocorrelation.

Definition 3.1.4

Let $Y = \{Y(\boldsymbol{x}) : \boldsymbol{x} \in X\}$ be a stochastic process.

(i) The function

$$c(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}) = \mathrm{cov}(Y(\boldsymbol{x}^{(1)}), Y(\boldsymbol{x}^{(2)}))$$

is called autocovariance function of Y.

(ii) The autocorrelation function of a process Y with autocovariance function c is

$$\begin{aligned} \rho(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}) &= \frac{c(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)})}{(\mathrm{var}(Y(\boldsymbol{x}^{(1)}))\mathrm{var}(Y(\boldsymbol{x}^{(2)})))^{1/2}} \\ &= \frac{\mathrm{cov}(Y(\boldsymbol{x}^{(1)}), Y(\boldsymbol{x}^{(2)})}{(\mathrm{var}(Y(\boldsymbol{x}^{(1)}))\mathrm{var}(Y(\boldsymbol{x}^{(2)})))^{1/2}} \\ &= \mathrm{cor}(Y(\boldsymbol{x}^{(1)}), Y(\boldsymbol{x}^{(2)})). \end{aligned}$$

Definition 3.1.5

(i) The real-valued process $Y = \{Y(\boldsymbol{x}) : \boldsymbol{x} \in X\}$ is called strongly stationary if the families $\{Y(\boldsymbol{x}^{(1)}), \ldots, Y(\boldsymbol{x}^{(n)})\}$ and $\{Y(\boldsymbol{x}^{(1)} + \boldsymbol{h}), \ldots, Y(\boldsymbol{x}^{(n)} + \boldsymbol{h})\}$ have the same joint distribution for all $\boldsymbol{h}$ and $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$ such that $\boldsymbol{x}^{(i)} \in X$ and $\boldsymbol{x}^{(i)} + \boldsymbol{h} \in X$, $i = 1, \ldots, n$. This means that the processes' finite-dimensional

distributions are invariant under shifts.
(ii) The real-valued process $Y = \{Y(\boldsymbol{x}) : \boldsymbol{x} \in X\}$ *is called weakly stationary if for all* $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)} \in X$ *and* $\boldsymbol{x}^{(1)} + \boldsymbol{h}, \boldsymbol{x}^{(2)} + \boldsymbol{h} \in X$

$$E(Y(\boldsymbol{x}^{(1)})) = E(Y(\boldsymbol{x}^{(2)})) \tag{3.1}$$

and

$$\operatorname{cov}(Y(\boldsymbol{x}^{(1)}), Y(\boldsymbol{x}^{(2)})) = \operatorname{cov}(Y(\boldsymbol{x}^{(1)} + \boldsymbol{h}), Y(\boldsymbol{x}^{(2)} + \boldsymbol{h})). \tag{3.2}$$

This means that the process has constant mean and its autocovariance is invariant under shifts.

It follows that a strongly stationary process is also weakly stationary.

Definition 3.1.6
If the vector of random variables $Y_x = (Y(\boldsymbol{x}^{(1)}), \ldots, Y(\boldsymbol{x}^{(n)}))^t$ *has joint density function*

$$p(\boldsymbol{y}) = ((2\pi)^n |C|)^{-1/2} \exp\left(-\frac{1}{2}(\boldsymbol{y} - \boldsymbol{\mu})C^{-1}(\boldsymbol{y} - \boldsymbol{\mu})^t\right)$$

where C *is a positive definite symmetric matrix, then* Y_x *is said to have the multivariate normal distribution* $N(\boldsymbol{\mu}(\boldsymbol{y}), C(\boldsymbol{y}))$.

Note that if Y_x has the multinormal distribution $N(\boldsymbol{\mu}, C)$ then $E(Y_x) = \boldsymbol{\mu}$, i.e. $E(Y(\boldsymbol{x}^{(i)})) = \mu_i$, and the matrix C is called the *covariance matrix* because $C_{ij} = \operatorname{cov}(Y(\boldsymbol{x}^{(i)}), Y(\boldsymbol{x}^{(j)})) = \operatorname{Cov}_{ij}$. Since $(Y_x - \boldsymbol{\mu})(Y_x - \boldsymbol{\mu})^t$ is a matrix with (i,j)th entry $(Y(\boldsymbol{x}^{(i)}) - \mu_i)(Y(\boldsymbol{x}^{(j)}) - \mu_j)$ and

$$E((Y(\boldsymbol{x}^{(i)}) - \mu_i)(Y(\boldsymbol{x}^{(j)}) - \mu_j)) = \operatorname{cov}(Y(\boldsymbol{x}^{(i)}), Y(\boldsymbol{x}^{(j)}))$$

the covariance matrix is often written as

$$C = E[(Y_x - \boldsymbol{\mu})(Y_x - \boldsymbol{\mu})^t].$$

Definition 3.1.7
A real-valued continuous process Y *is called Gaussian if for each finite-dimensional* $(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)})$ *the corresponding random vector* $Y_x = (Y(\boldsymbol{x}^{(1)}), \ldots, Y(\boldsymbol{x}^{(n)}))$ *has the multivariate normal distribution* $N(\boldsymbol{\mu}, C)$ *for a mean vector* $\boldsymbol{\mu}$ *and a covariance matrix* C.

The joint density function of a Gaussian stochastic process is completely determined by its mean and covariance. Therefore a Gaussian stochastic process is strongly stationary if and only if it is weakly stationary, and we will call a Gaussian stochastic process with this property *stationary*. By definition,

a stationary Gaussian stochastic process has constant mean vector such that $\boldsymbol{\mu} = E(Y(\boldsymbol{x}))\mathbf{1}$ for any $\boldsymbol{x} \in X$, by (3.1), and it has constant process variance, $\text{var}(Y(\boldsymbol{x})) = \text{cov}(Y(\boldsymbol{x}), Y(\boldsymbol{x})) = \text{cov}(Y(\boldsymbol{v}), Y(\boldsymbol{v})) = \text{var}(Y(\boldsymbol{v}))$ for any $\boldsymbol{x}, \boldsymbol{v} \in X$, by (3.2). We will also denote this variance by σ^2.

Definition 3.1.8

(i) The scalar random variable Z is said to have the gamma distribution with parameters $\lambda, q > 0$, if it has density function

$$f(z) = \frac{1}{\int_0^\infty z^{q-1}e^{-z}dz}\lambda^q z^{q-1}e^{-\lambda z}, \quad z \geq 0.$$

(ii) Let $N_1, \ldots, N_u$ be independent $N(0,1)$ variables and let

$$. \quad Z = N_1^2 + \ldots + N_u^2.$$

Then Z is said to have chi-squared distribution, denoted by χ^2, with u degrees of freedom. The density function for χ^2 with u degrees of freedom is given by the gamma distribution with $\lambda = \frac{1}{2}$ and $q = \frac{u}{2}$, i.e. Z has probability density function

$$f(z) = \frac{1}{\int_0^\infty z^{\frac{u}{2}-1}e^{-z}dz}\left(\frac{1}{2}\right)^{\frac{u}{2}} z^{\frac{u}{2}-1}e^{-\frac{1}{2}z}, \quad z \geq 0.$$

Definition 3.1.9

We define $G(\delta, k^2)$ as a probability distribution with parameters δ and k with density function

$$g(v, \delta, k^2) = K_\delta k^\delta v^{\delta-1}\exp\left(-\frac{k^2v^2}{2}\right),$$

for some suitable normalizing constant K_δ.

The G distribution is related to the χ^2 distribution as follows: let

$$w = (kv)^2 \tag{3.3}$$

then

$$K_\delta k^\delta v^{\delta-1}\exp\left(-\frac{(kv)^2}{2}\right)dv = \frac{1}{2}K_\delta w^{\frac{\delta}{2}-1}\exp\left(-\frac{w}{2}\right)dw.$$

This is a χ^2 distribution with δ degrees of freedom. To generate χ^2 random deviates with δ degrees of freedom, the following method can be used: if δ is a small even integer, the random deviate w can be obtained from $\delta/2$ uniform $U(0,1)$ random deviates U_i by

$$w = -2\log\left(\prod_{i=1}^{\delta/2} U_i\right). \tag{3.4}$$

If δ is odd, the random deviate w can be obtained by

$$w = -2\log\left(\prod_{i=1}^{(\delta-1)/2} U_i\right) + N^2, \tag{3.5}$$

where N has standard normal distribution, i.e. $N \sim N(0,1)$. This method can be found for example in Gentle's book [25]. Then to obtain a random sample v from the $G(\delta, k)$ distribution we generate a w using (3.4) or (3.5) and then calculate v using (3.3).

3.2 Derivation of the BLUP and the MSE

The expected improvement criterion, which is examined later on in this thesis, is based on an interpolating approximation to the objective function and an error function. The derivation of these is given in this section. This can also be found for example in papers by Sacks *et al.* [60, 61] and Schonlau's thesis [63].

The approach described here is based on the Kriging model approach used in Geostatistics. The Kriging model consists of two components. The first component is a general linear model, the second component is the departure from the linear model which is treated as the realization of a stationary Gaussian stochastic process. Let $D = [0,1]^d$ denote the design or sample space, let $\boldsymbol{x} \in D$ be a (scaled) vector of input values. The Kriging approach models the response as

$$Y(\boldsymbol{x}) = \sum_{j=1}^{m} \beta_j f_j(\boldsymbol{x}) + \epsilon(\boldsymbol{x}), \tag{3.6}$$

where $\epsilon(\boldsymbol{x})$ is a stationary Gaussian stochastic process with expectation $E(\epsilon(\boldsymbol{x})) = 0$, process variance σ^2, and covariance $\mathrm{cov}(\epsilon(\boldsymbol{x}), \epsilon(\overline{\boldsymbol{x}})) = \sigma^2 \mathrm{cor}(\epsilon(\boldsymbol{x}), \epsilon(\overline{\boldsymbol{x}}))$ for input vectors $\boldsymbol{x}$, $\overline{\boldsymbol{x}}$ and a correlation function $\mathrm{cor}(.,.)$, and where $(f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))^t$ are m known (regression) functions. We consider the case where the functions $(f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))^t$ form a polynomial basis. For any $\boldsymbol{x} \in D$ the output $y(\boldsymbol{x})$ is a sample path or realization of a Gaussian stochastic process with expectation $\sum \beta_j f_j(\boldsymbol{x})$ and variance σ^2. There are different possible choices of correlation functions, some of which were mentioned in Chapter 2. A generalized exponential autocorrelation function ρ_K is used here, which is defined by

$$\rho_K(\boldsymbol{x}, \overline{\boldsymbol{x}}) = \prod_{j=1}^{d} \exp(-\theta_j |x_j - \overline{x}_j|^{p_j}) \tag{3.7}$$

where $\theta_j > 0$ and $0 < p_j \leq 2$. For details of this approach see for example the papers by Sacks *et al.* [60, 61], and Schonlau's thesis [63]. The $\boldsymbol{p}$ and $\boldsymbol{\theta}$ influence

the smoothness of the response surface and how global or local the estimator is. Larger θ_j will make correlation smaller in general. The p_j are smoothness parameters. Note that if derivatives are to be used, the correlation function has to be at least twice differentiable, that is $p_j = 2$ for $j = 1, \ldots, d$. Details of conditions for differentiability can be found in [54]. The autocorrelation function ρ_K controls properties of $\epsilon(.)$. In particular, with the above definition (3.7) the correlation $\rho_K(\boldsymbol{x}, \overline{\boldsymbol{x}})$ and the covariance $\mathrm{cov}(\epsilon(\boldsymbol{x}), \epsilon(\overline{\boldsymbol{x}}))$ are functions of $|x_j - \overline{x}_j|$ only, $j = 1, \ldots, d$. The correlation will be near one when the distance between two points is small and it will be near zero for points that are a large distance apart, depending on the values of the θ_j.

Suppose we have n d-dimensional vectors $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)} \in D$ at which the response $y(\boldsymbol{x})$ has been evaluated and the corresponding response values are $\boldsymbol{y} = (y_1, \ldots, y_n)^t$. Let R denote the matrix $(\rho_K(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}))_{1 \le i,j \le n}$ and let $\boldsymbol{r}(\boldsymbol{x})$ or $\boldsymbol{r}_x$ denote the vector $(\rho_K(\boldsymbol{x}, \boldsymbol{x}^{(i)}))_{1 \le i \le n}$, so that

$$r_i(\boldsymbol{x}) = \prod_{j=1}^{d} \exp(-\theta_j |x_j - x_j^{(i)}|^{p_j}), \tag{3.8}$$

and $\boldsymbol{r}_x$ is the vector of correlations between a previously unsampled point $\boldsymbol{x}$ and the sample points $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$. The following notation will be used,

$$\begin{aligned} \boldsymbol{f}_x &= (f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))^t \\ F &= (f_j(\boldsymbol{x}^{(i)}))_{\substack{i=1,\ldots,n \\ j=1,\ldots,m}}, \end{aligned}$$

thus F denotes the $n \times m$ matrix of m polynomial basis elements evaluated at n sample points. For the data $Y = (Y_1, \ldots, Y_n)^t$ the above Gaussian stochastic process model (3.6) looks as follows,

$$Y = F\boldsymbol{\beta} + \boldsymbol{\epsilon}(\boldsymbol{x})$$

where $\boldsymbol{\epsilon}(\boldsymbol{x})$ is the departure from the linear model $\boldsymbol{\epsilon}(\boldsymbol{x}) = Y - F\boldsymbol{\beta}$, and it has expectation $E(\boldsymbol{\epsilon}(\boldsymbol{x})) = E(Y) - F\boldsymbol{\beta} = \boldsymbol{0}$. Since $\boldsymbol{\epsilon}(\boldsymbol{x})$ is a stationary Gaussian stochastic process, the $\epsilon_i(\boldsymbol{x})$ have the same variance σ^2. The expectation and covariance of Y are, respectively

$$\begin{aligned} E(Y) &= F\boldsymbol{\beta} \\ \mathrm{cov}(\mathrm{Y}(\boldsymbol{x}), \mathrm{Y}(\overline{\boldsymbol{x}})) &= \sigma^2 \rho_K(\boldsymbol{x}, \overline{\boldsymbol{x}}). \end{aligned}$$

Consider a linear predictor $\boldsymbol{k}_x^t \boldsymbol{y}$ of $Y(\boldsymbol{x})$. The best linear unbiased predictor (BLUP) finds the $\boldsymbol{k}_x$ that minimizes the mean squared error (MSE) of prediction subject to the unbiasedness constraint. The MSE is given by

$$
\begin{aligned}
s^2[\boldsymbol{k}_x^t\boldsymbol{y}] &= E[\boldsymbol{k}_x^t\boldsymbol{y} - Y(\boldsymbol{x})]^2 \\
&= (E[\boldsymbol{k}_x^t\boldsymbol{y} - Y(\boldsymbol{x})])^2 + \mathrm{var}(\boldsymbol{k}_x^t\boldsymbol{y} - Y(\boldsymbol{x})) \\
&= (\boldsymbol{k}_x^t F\boldsymbol{\beta} - \boldsymbol{f}_x^t\boldsymbol{\beta})^2 + \mathrm{var}(\boldsymbol{k}_x^t\boldsymbol{y}) - \mathrm{cov}(\boldsymbol{k}_x^t\boldsymbol{y}, Y(\boldsymbol{x})) \\
&\quad -\mathrm{cov}(Y(\boldsymbol{x}), \boldsymbol{k}_x^t\boldsymbol{y}) + \mathrm{var}(Y(\boldsymbol{x})) \\
&= (\boldsymbol{k}_x^t F\boldsymbol{\beta} - \boldsymbol{f}_x^t\boldsymbol{\beta})^2 + \boldsymbol{k}_x^t\mathrm{var}(\boldsymbol{y})\boldsymbol{k}_x - \boldsymbol{k}_x^t\mathrm{cov}(\boldsymbol{y}, Y(\boldsymbol{x})) \\
&\quad -\mathrm{cov}(Y(\boldsymbol{x}), \boldsymbol{y})\boldsymbol{k}_x + \sigma^2 \\
&= (\boldsymbol{k}_x^t F\boldsymbol{\beta} - \boldsymbol{f}_x^t\boldsymbol{\beta})^2 + \boldsymbol{k}_x^t\sigma^2 R\boldsymbol{k}_x - 2\boldsymbol{k}_x^t\sigma^2\boldsymbol{r}_x + \sigma^2. \qquad (3.9)
\end{aligned}
$$

Equating $E(\boldsymbol{k}_x^t\boldsymbol{y}) = \boldsymbol{k}_x^t F\boldsymbol{\beta}$ and $E(Y) = \boldsymbol{f}_x^t\boldsymbol{\beta}$ gives the unbiasedness constraint

$$\boldsymbol{k}_x^t F = \boldsymbol{f}_x^t,$$

this makes $\boldsymbol{k}_x^t\boldsymbol{y}$ an unbiased estimate of Y. With this unbiasedness constraint expression (3.9) simplifies to

$$s^2[\boldsymbol{k}_x^t\boldsymbol{y}] = \boldsymbol{k}_x^t\sigma^2 R\boldsymbol{k}_x + \sigma^2 - 2\boldsymbol{k}_x^t\sigma^2\boldsymbol{r}_x. \qquad (3.10)$$

To find the BLUP we have to solve the following minimization problem

$$\begin{aligned}
&\text{minimize} \quad s^2[\boldsymbol{k}_x^t\boldsymbol{y}] && (3.11)\\
&s.t. \qquad \boldsymbol{k}_x^t F = \boldsymbol{f}_x^t. && (3.12)
\end{aligned}$$

Scaling the constraint (3.12) and introducing Lagrange multipliers the problem takes the form

$$\text{minimize} \quad \boldsymbol{k}_x^t\sigma^2 R\boldsymbol{k}_x - 2\boldsymbol{k}_x^t\sigma^2\boldsymbol{r}_x + \sigma^2 - 2\sigma^2\boldsymbol{\lambda}^t(F^t\boldsymbol{k}_x - \boldsymbol{f}_x).$$

Taking derivatives yields the necessary first-order optimality conditions for k_x to be an optimum for problem (3.11)–(3.12),

$$\begin{aligned}
R\boldsymbol{k}_x - \boldsymbol{r}_x - F\boldsymbol{\lambda} &= \mathbf{0} \\
\boldsymbol{k}_x^t F - \boldsymbol{f}_x^t &= \mathbf{0}
\end{aligned}$$

which corresponds to the following system,

$$\begin{pmatrix} 0 & F^t \\ F & R \end{pmatrix}\begin{pmatrix} -\boldsymbol{\lambda} \\ \boldsymbol{k}_x \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}_x \\ \boldsymbol{r}_x \end{pmatrix}. \qquad (3.13)$$

The matrix

$$\begin{pmatrix} 0 & F^t \\ F & R \end{pmatrix}$$

is non-singular, and therefore invertible, under the following conditions: the matrix R is positive definite, $m \leq n$, and F has full rank m. The matrix F of polynomial basis functions has full row rank m if the number of sample points n is greater than the maximum degree of any of the basis functions. In our case the matrix R is always positive definite and we assume that the number of sample points exceeds the maximum degree of any of the basis functions. Consequently, there exists a unique solution to the necessary optimality conditions. The BLUP is then $\hat{y} = \boldsymbol{k}_x^t \boldsymbol{y}$ for the $\boldsymbol{k}_x$ that solves the above problem (3.11)–(3.12), so

$$\begin{aligned} \hat{y}(\boldsymbol{x}) = \boldsymbol{k}_x^t \boldsymbol{y} &= (-\boldsymbol{\lambda}^t, \boldsymbol{k}_x^t) \begin{pmatrix} \mathbf{0} \\ \boldsymbol{y} \end{pmatrix} \\ &= (\boldsymbol{f}_x^t, \boldsymbol{r}_x^t) \begin{pmatrix} 0 & F^t \\ F & R \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{0} \\ \boldsymbol{y} \end{pmatrix} \end{aligned}$$

which can be written as

$$\hat{y}(\boldsymbol{x}) = \boldsymbol{f}_x^t \hat{\boldsymbol{\beta}} + \boldsymbol{r}_x^t R^{-1}(\boldsymbol{y} - F\hat{\boldsymbol{\beta}})$$

where

$$\hat{\boldsymbol{\beta}} = (F^t R^{-1} F)^{-1} F^t R^{-1} \boldsymbol{y}.$$

To see this note that

$$\begin{pmatrix} 0 & F^t \\ F & R \end{pmatrix} \begin{pmatrix} \hat{\boldsymbol{\beta}} \\ R^{-1}(\boldsymbol{y} - F\hat{\boldsymbol{\beta}}) \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \boldsymbol{y} \end{pmatrix}.$$

Note that the correlation matrix R is positive definite, as is shown in Chapter 5. It is therefore invertible, and it can be factorized in Cholesky factors which can be used to solve any systems involving the matrix R. The $\hat{\boldsymbol{\beta}}$ is the generalized least squares and also the maximum likelihood estimate of $\boldsymbol{\beta}$ as in (3.6). Most commonly only a trivial regression function is used, $f = 1$. This gives a constant regression term and in this case the BLUP simplifies to

$$\hat{y}(\boldsymbol{x}) = \hat{\mu} + \boldsymbol{r}_x^t R^{-1}(\boldsymbol{y} - \mathbf{1}\hat{\mu})$$

where $\hat{\mu}$ is the generalized least squares and the maximum likelihood estimator of μ,

$$\hat{\mu} = (\mathbf{1}^t R^{-1} \mathbf{1})^{-1} \mathbf{1}^t R^{-1} \boldsymbol{y}.$$

At any design point $\boldsymbol{x}^{(i)}$, $i = 1, \ldots, n$ the predictor $\hat{y}$ satisfies

$$\begin{aligned} \hat{y}(\boldsymbol{x}^{(i)}) &= \boldsymbol{f}_x^t \hat{\boldsymbol{\beta}} + \boldsymbol{e}^{(i)}(\boldsymbol{y} - F\hat{\boldsymbol{\beta}}) \\ &= y_i, \end{aligned}$$

where $\boldsymbol{e}_i$ is the n-vector with 1 in position i and 0 everywhere else. So $\hat{y}$ interpolates the objective function in $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$.

To derive a formula for the mean squared error we consider the above expressions for the MSE s^2 (3.10) and the system (3.13). From (3.13) it follows that

$$\boldsymbol{k}_x^t R \boldsymbol{k}_x - 2\boldsymbol{k}_x^t \boldsymbol{r}_x = \boldsymbol{k}_x^t F \boldsymbol{\lambda} - \boldsymbol{k}_x^t \boldsymbol{r}_x = \boldsymbol{f}_x^t \boldsymbol{\lambda} - \boldsymbol{k}_x^t \boldsymbol{r}_x,$$

and we can derive the following expression for the mean squared error,

$$\begin{aligned}
s^2(\hat{y}(\boldsymbol{x})) &= E[\boldsymbol{k}_x^t \boldsymbol{y} - Y(\boldsymbol{x})]^2 \\
&= \sigma^2 + \boldsymbol{k}_x^t \sigma^2 R \boldsymbol{k}_x - 2\boldsymbol{k}_x^t \sigma^2 \boldsymbol{r}_x \\
&= \sigma^2 [1 + \boldsymbol{k}_x^t R \boldsymbol{k}_x - 2\boldsymbol{k}_x^t \boldsymbol{r}_x] \\
&= \sigma^2 \left[1 - (\boldsymbol{f}_x^t, \boldsymbol{r}_x^t) \begin{pmatrix} -\boldsymbol{\lambda} \\ \boldsymbol{k}_x \end{pmatrix} \right] \qquad (3.14) \\
&= \sigma^2 \left[1 - (\boldsymbol{f}_x^t, \boldsymbol{r}_x^t) \begin{pmatrix} 0 & F^t \\ F & R \end{pmatrix}^{-1} \begin{pmatrix} \boldsymbol{f}_x \\ \boldsymbol{r}_x \end{pmatrix} \right]. \qquad (3.15)
\end{aligned}$$

It is not difficult to see that

$$\begin{pmatrix} 0 & F^t \\ F & R \end{pmatrix} \begin{pmatrix} -(F^t R^{-1} F)^{-1} \boldsymbol{f}_x + (F^t R^{-1} F)^{-1} F^t R^{-1} \boldsymbol{r}_x \\ R^{-1} F (F^t R^{-1} F)^{-1} \boldsymbol{f}_x + R^{-1} \boldsymbol{r}_x - R^{-1} F (F^t R^{-1} F)^{-1} F^t R^{-1} \boldsymbol{r}_x \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}_x \\ \boldsymbol{r}_x \end{pmatrix},$$

and using this expression, (3.15) can be rewritten as

$$\begin{aligned}
s^2(\hat{y}(\boldsymbol{x})) &= \sigma^2 [1 - \boldsymbol{r}_x^t R^{-1} \boldsymbol{r}_x + \boldsymbol{f}_x^t (F^t R F)^{-1} \boldsymbol{f}_x - 2\boldsymbol{r}_x^t R^{-1} F (F^t R F)^{-1} \boldsymbol{f}_x \\
&\quad + \boldsymbol{r}_x^t R^{-1} F (F^t R F)^{-1} F^t R^{-1} \boldsymbol{r}_x]. \qquad (3.16)
\end{aligned}$$

In the case where $f = 1$ and $F = \mathbf{1}$ this simplifies to

$$s^2(\hat{y}(\boldsymbol{x})) = \sigma^2 \left[1 - \boldsymbol{r}_x^t R^{-1} \boldsymbol{r}_x + \frac{(1 - \mathbf{1}^t R^{-1} \boldsymbol{r}_x)^2}{\mathbf{1}^t R^{-1} \mathbf{1}} \right]. \qquad (3.17)$$

At any design point $\boldsymbol{x}^{(i)}$, $i = 1, \ldots, n$ the mean squared error satisfies

$$s^2(\hat{y}(\boldsymbol{x}^{(i)})) = 0.$$

This can be shown by substituting $R^{-1} \boldsymbol{r}_{x^{(i)}} = \boldsymbol{e}^{(i)}$ and $F^t \boldsymbol{e}^{(i)} = \boldsymbol{f}_{x^{(i)}}$ into (3.16).

We have now found a linear predictor $\hat{y}(\boldsymbol{x})$ of $Y(\boldsymbol{x})$, and an estimate $s^2(\boldsymbol{x})$ of the error of this linear predictor. At every point $\boldsymbol{x} \in [0,1]^d$, $\hat{y}(\boldsymbol{x})$ can be interpreted as an expected value of the objective function $y(\boldsymbol{x})$ at that point, given the known data points $(\boldsymbol{x}^{(1)}, y_1), \ldots, (\boldsymbol{x}^{(n)}, y_n)$. The MSE $s^2(\boldsymbol{x})$ is an estimate of how good or bad an estimate this expected value is. For fixed $\boldsymbol{\theta}$ and σ^2, $s^2(\boldsymbol{x})$ only depends on the distance of $\boldsymbol{x}$ from any previous sample points, $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$. At the previous sample points, $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$, the objective function values are known, and there is no uncertainty about the BLUP. At these points the MSE is 0.

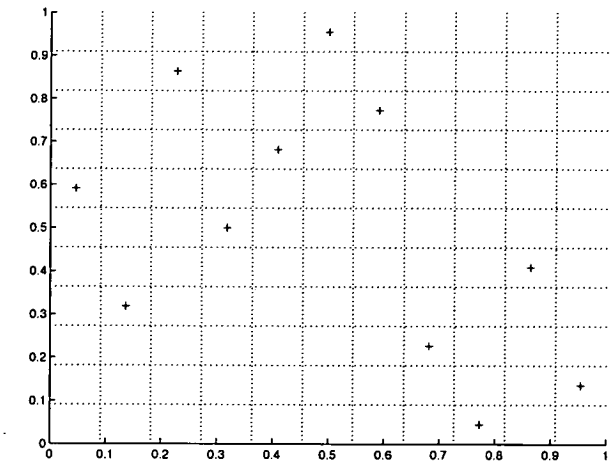

Figure 3.1: Latin Hypercube with 11 points.

3.3 Initial Sample

Minimizing the objective function requires choosing a set of n initial sample points in the sample space $D = [0, 1]^d$, also called an initial design, at which the objective function is to be evaluated to start with. As the optimization proceeds, more points will be added according to a sampling criterion, but no such criterion is available for the initial points, and they are often chosen from a so-called experimental design. There are a number of different experimental designs that can be used to find an initial design. Koehler and Owen for example in [38] look at different designs, variations of them, and their properties. We use a common variant of Latin hypercube sampling, also called Latin hypercube sampling with centered points, or lattice samples, to find the initial design with n points. In d variables or dimensions and with n sample points the points are found by

$$x_j^{(i)} = \frac{\pi^{(j)}(i) - 0.5}{n}$$

where $\pi^{(j)}$, $j = 1, \ldots, d$ are independent uniform random permutations of the integers $1, \ldots, n$. By this design the range $[0, 1]$ of any input variable x_j is split into n intervals of the same length and the Latin hypercube sample projected onto any one of these input variables has exactly one point at the centre of any of these intervals, and therefore each of the input variables is explored in each of the n intervals. Figure 3.1 shows an example of a Latin hypercube with $n = 11$ centered points in 2 dimensions.

Latin Hypercubes as initial designs have the advantage that besides having a random element, every dimension is explored in the sense described above. Advantage over regular grids are particularly relevant when working in higher

dimensions, where the number of points sampled on a regular grid can be prohibitively large; this advantage is particularly prominent if the function depends very strongly on few input variables. At the same time the design space is explored better in every variable than if the choice of initial points were made by uniform sampling over the hyperrectangle.

3.4 Parameter Estimation

The model has parameters which have to be estimated. This could be parameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_d)$ and $\boldsymbol{p} = (p_1, \ldots, p_d)$, or parameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_d)$, if the p_i are fixed. Once these have been found, the estimates $\hat{\mu}$ or $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}$ can be computed. Two popular methods for parameter estimation are introduced here, Maximum Likelihood Estimation (MLE) and Cross Validation (CV).

Both of these methods give a single point estimate of the parameters, in contrast to Bayesian parameter estimation where the entire posterior distribution of the parameters is taken into account.

3.4.1 Maximum Likelihood Estimation

By using maximum likelihood estimation we find the parameters for which the given set of observations has maximum probability. With observations $\boldsymbol{y} = (y_1, \ldots, y_n)$, a random sample from the normal distribution with mean $F\boldsymbol{\beta}$ and variance-covariance matrix $\sigma^2 R$, the likelihood function is

$$L(\boldsymbol{y}) = |2\pi\sigma^2 R|^{-1/2} \exp\left[\left(-\frac{1}{2}\right)(\boldsymbol{y} - F\boldsymbol{\beta})^t \sigma^{-2} R^{-1} (\boldsymbol{y} - F\boldsymbol{\beta})\right]. \qquad (3.18)$$

Often it is more convenient to use the log-likelihood function instead. This is

$$\begin{aligned} \ln L(\boldsymbol{y}) &= \ln\left(|2\pi\sigma^2 R|^{-1/2}\right) - \frac{1}{2}(\boldsymbol{y} - F\boldsymbol{\beta})^t \sigma^{-2} R^{-1} (\boldsymbol{y} - F\boldsymbol{\beta}) \\ &= -\frac{n}{2}\ln\left(2\pi\sigma^2\right) - \frac{1}{2}\ln\left(|R|\right) - \frac{1}{2}(\boldsymbol{y} - F\boldsymbol{\beta})^t \sigma^{-2} R^{-1} (\boldsymbol{y} - F\boldsymbol{\beta}) \\ &= -\frac{1}{2}[n\ln\left(2\pi\sigma^2\right) + \ln\left(|R|\right) + (\boldsymbol{y} - F\boldsymbol{\beta})^t \sigma^{-2} R^{-1} (\boldsymbol{y} - F\boldsymbol{\beta})]. \qquad (3.19) \end{aligned}$$

Assuming that $\boldsymbol{\theta}$ and $\boldsymbol{p}$ are known, differentiating with respect to $\boldsymbol{\beta}$ and σ^2 and setting the derivatives to zero gives the maximum likelihood estimate (MLE) of $\boldsymbol{\beta}$ and of σ^2, respectively,

$$\begin{aligned} \hat{\boldsymbol{\beta}} &= \left(F^t R^{-1} F\right)^{-1} F^t R^{-1} \boldsymbol{y} \\ \hat{\sigma}^2 &= \frac{1}{n}(\boldsymbol{y} - F\boldsymbol{\beta})^t R^{-1} (\boldsymbol{y} - F\boldsymbol{\beta}). \end{aligned}$$

Substituting $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}^2$ into (3.19) we obtain

$$\begin{aligned} \ln L(\boldsymbol{y}) &= -\frac{1}{2}[n \ln(\hat{\sigma}^2) + \ln(|R|)] - \frac{1}{2}[n \ln(2\pi) + n] \\ &= -\frac{1}{2}[n \ln(\hat{\sigma}^2) + \ln(|R|)] + \text{const} \end{aligned}$$

as the function to be maximized to find the maximum likelihood estimates of $\boldsymbol{\theta}$ and $\boldsymbol{p}$. See for example Schonlau's thesis [63] and for further background reading the book by Mardia *et al.* [46] or Seber [68].

3.4.2 Cross Validation

How well does the BLUP function $\hat{y}(\boldsymbol{x})$ agree with our observations? For given parameters we compare the observations of the objective function with the values of the BLUP at the respective sample points. One of the n points which we have already sampled at, $\boldsymbol{x}^{(i)}$ say, is left out of the sample and the BLUP for the function value at that point is found using the remaining $n-1$ points, this is denoted by $\hat{y}_{-i}$. We calculate the BLUP as a function of the parameters $\theta_1, \ldots, \theta_d, p_1, \ldots, p_d$, evaluate the sum of squares of the differences between the actual function values and the corresponding values of $\hat{y}$ and minimize this sum with respect to $\theta_1, \ldots, \theta_d$ and $p_1, \ldots, p_d$

$$\min \sum_{i=1}^{n} (\hat{y}_{-i} - y_i)^2.$$

Cross validation can be used to validate the model, see for example the paper by Schonlau *et al.* [64], or to estimate the model parameters. The problem with cross validation here is that for every iteration in the optimization routine, $\hat{y}$ has to be evaluated n times and each of these evaluations involves inverting matrices and is expensive for large n. For the estimation of the model parameters we shall prefer maximum likelihood estimation.

3.5 Expected Improvement

While we want to sample the function at points where the expected function value $\hat{y}$, the BLUP, is low, we also want to keep sampling at points far from previous sampling points, where the uncertainty in the function value s^2, the MSE, is high. This aim can be quantified by the expected improvement function, and by maximizing it we find new points where we expect maximal improvement in the function values. With the two different aspects of sampling where $\hat{y}$ is low and

s^2 is high, expected improvement (EI) is a way of balancing global versus local search.

We only know a few points of our function $y(\boldsymbol{x})$ that is to be minimized. We model the uncertainty in $y(\boldsymbol{x})$ at a point $\boldsymbol{x} \in D$ by treating it as the realization of a normally distributed random variable $Y(\boldsymbol{x})$ with mean and standard deviation given by the BLUP $\hat{y}$ and $s = \sqrt{\text{MSE}}$. So $Y(\boldsymbol{x})$ is $N(\hat{y}(\boldsymbol{x}), s^2(\boldsymbol{x}))$ distributed and $\dfrac{Y(\boldsymbol{x}) - \hat{y}(\boldsymbol{x})}{s(\boldsymbol{x})}$ is $N(0,1)$ distributed. Let y_{0n} be the best function value found up to this point, then the improvement at the point $\boldsymbol{x}$ is

$$I(\boldsymbol{x}) = \max(y_{0n} - Y, 0).$$

The expected improvement is

$$\begin{aligned} E[I(\boldsymbol{x})] &= E[\max(y_{0n} - Y, 0)] \\ &= \int_{-\infty}^{y_{0n}} (y_{0n} - y)\phi_x(y)dy \end{aligned}$$

where $\phi_x(.)$ is the probability density function of $Y(\boldsymbol{x})$. This can be written as

$$E[I(\boldsymbol{x})] = (y_{0n} - \hat{y})\Phi\left(\frac{y_{0n} - \hat{y}}{s}\right) + s\phi\left(\frac{y_{0n} - \hat{y}}{s}\right)$$

where $\phi(.)$ and $\Phi(.)$ are the standard normal probability density and the cumulative distribution function respectively. Jones *et al.* in [34] state an interesting property of the expected improvement. This property becomes obvious when taking the derivatives of $E(I)$ with respect to s and $\hat{y}$. This gives

$$\begin{aligned} \frac{\partial E(I)}{\partial \hat{y}} &= -\Phi\left(\frac{y_{0n} - \hat{y}}{s}\right) - (y_{0n} - \hat{y})\left(\frac{1}{s}\right)\phi\left(\frac{y_{0n} - \hat{y}}{s}\right) \\ &\qquad + s\left(\frac{y_{0n} - \hat{y}}{s^2}\right)\phi\left(\frac{y_{0n} - \hat{y}}{s}\right) \\ &= -\Phi\left(\frac{y_{0n} - \hat{y}}{s}\right) < 0 \end{aligned} \tag{3.20}$$

and

$$\begin{aligned} \frac{\partial E(I)}{\partial s} &= (y_{0n} - \hat{y})\left(-\frac{y_{0n} - \hat{y}}{s^2}\right)\phi\left(\frac{y_{0n} - \hat{y}}{s}\right) \\ &\qquad + \phi\left(\frac{y_{0n} - \hat{y}}{s}\right) + s\left(\frac{(y_{0n} - \hat{y})^2}{s^3}\right)\phi\left(\frac{y_{0n} - \hat{y}}{s}\right) \\ &= \phi\left(\frac{y_{0n} - \hat{y}}{s}\right) > 0. \end{aligned} \tag{3.21}$$

Hence $E(I)$ is monotonic in s and $\hat{y}$, more specifically strictly monotonically increasing in s and strictly monotonically decreasing in $\hat{y}$.

The expected improvement is now used as a criterion for finding new sample points at which the original function is to be evaluated. The expected improvement function is multimodal and maximizing it to global optimality therefore requires a global optimization technique. It should be observed that all merit functions which include an error term will be multimodal and therefore require global optimization. This global optimization can be achieved by branch and bound, to which we will now give a short introduction before returning to the problem of applying it to the expected improvement function.

3.6 Branch and Bound

Generally in branch and bound the original problem is divided into a sequence of subproblems which are solved to give the solution to the original problem. Since we use branch and bound only for maximizing we will explain it only in the maximization context. Branching is done by the bound constraints. We start with the initial box, here $D = [0, 1]^d$, and find a lower and an upper bound for the maximum of the objective function, here the expected improvement function, on this box. To find a lower bound on the maximum we can use any value of the objective function on that box. The box is then split into two smaller boxes in the following way: it is split in half along the longest side and if there is more than one side with the same length we choose the one with the lowest index. One of these boxes, for example the box with the lower half of the side we have just split, is kept to work on while the other box is stored on a waiting list. Alternatively the two boxes are put onto the waiting list and the box on the waiting list with the highest upper bound on the objective function is used as the new working box — this corresponds to best-first search, rather than depth-first search, and best-first search is the method we use. Now a lower and an upper bound for the maximum of the objective function on the new box are found. If the upper bound on the box is lower than the largest previously found lower bound, then that box is discarded as nonpromising and a new box to work on is taken from the waiting list. So in brief, if we are working on a box, we split the box, choose a box to work on, find the bounds, either discard the box to be replaced by a box from the waiting list or keep it, start again by splitting it etc. If there is more than one box on the waiting list we choose the one with the highest upper bound. If there are several with the same upper bound we choose the one nearest the bottom of the list. Thus we create a binary tree of boxes or subproblems and if it becomes clear that one node of the tree does not yield the optimum, we do not look into the rest of the branch any further but work on a different one instead.

For details on branch and bound methods see for example the book by Nemhauser and Wolsey [51].

3.7 Maximizing the Expected Improvement

By maximizing the expected improvement we find new promising points to evaluate the original function at. One way of maximizing the expected improvement function is by branch and bound. For details see for example the paper by Jones *et al.* [34], this is the approach that is followed in this section. Maximizing the expected improvement by branch and bound requires finding upper bounds on the expected improvement on the boxes. By the above stated monotonicity properties (3.20) and (3.21) these can be found by maximizing $s^2(\boldsymbol{x})$ and minimizing $\hat{y}(\boldsymbol{x})$. Here $s^2(\boldsymbol{x})$ and $\hat{y}(\boldsymbol{x})$ are regarded as functions of the variables $\boldsymbol{x}$ and $\boldsymbol{r}$ and the relation between $\boldsymbol{x}$ and $\boldsymbol{r}$ is fixed in the constraints. So we have the following two problems

Problem 1:

$$\begin{aligned} \min \quad & \hat{y}(\boldsymbol{x}, \boldsymbol{r}_x) = \boldsymbol{f}_x^t \hat{\boldsymbol{\beta}} + \boldsymbol{r}_x^t R^{-1}(\boldsymbol{y} - F\hat{\boldsymbol{\beta}}) \\ s.t. \quad & r_i - \prod_{j=1}^{d} \exp(-\theta_j |x_j - x_j^{(i)}|^{p_j}) = 0 \qquad i = 1, \ldots, n \\ & x_j^L \le x_j \le x_j^U \qquad j = 1, \ldots, d \end{aligned} \tag{3.22}$$

Problem 2:

$$\begin{aligned} \max \quad & s^2(\boldsymbol{x}, \boldsymbol{r}_x) = \sigma^2[1 - \boldsymbol{r}_x^t R^{-1} \boldsymbol{r}_x + \boldsymbol{f}_x^t (F^t R F)^{-1} \boldsymbol{f}_x - 2\boldsymbol{r}_x^t R^{-1} F (F^t R F)^{-1} \boldsymbol{f}_x \\ & \quad + \boldsymbol{r}_x^t R^{-1} F (F^t R F)^{-1} F^t R^{-1} \boldsymbol{r}_x] \\ s.t. \quad & r_i - \prod_{j=1}^{d} \exp(-\theta_j |x_j - x_j^{(i)}|^{p_j}) = 0 \qquad i = 1, \ldots, n \\ & x_j^L \le x_j \le x_j^U \qquad j = 1, \ldots, d. \end{aligned} \tag{3.23}$$

The lower bounds x_j^L and the upper bounds x_j^U on the x_j give lower and upper bounds on the r_i. So we get the additional bound constraints $r_i^L \le r_i \le r_i^U$, $i = 1, \ldots, n$. Rather than maximizing s^2 we will look at the equivalent problem of minimizing $-s^2$. The two problems to be solved now are the following

Problem 1:

$$\begin{aligned} \min \quad & \hat{y}(\boldsymbol{x}, \boldsymbol{r}_x) = \boldsymbol{f}_x^t \hat{\boldsymbol{\beta}} + \boldsymbol{r}_x^t R^{-1}(\boldsymbol{y} - F\hat{\boldsymbol{\beta}}) \\ s.t. \quad & r_i - \prod_{j=1}^{d} \exp(-\theta_j |x_j - x_j^{(i)}|^{p_j}) = 0 \qquad i = 1, \ldots, n \\ & x_j^L \le x_j \le x_j^U \qquad j = 1, \ldots, d \\ & r_i^L \le r_i \le r_i^U \qquad i = 1, \ldots, n \end{aligned}$$

Problem 2:

$$\begin{aligned}
\min \quad & -s^2(\boldsymbol{x}, \boldsymbol{r}_x) = -\sigma^2[1 - \boldsymbol{r}_x^t R^{-1} \boldsymbol{r}_x + \boldsymbol{f}_x^t (F^t R F)^{-1} \boldsymbol{f}_x - 2\boldsymbol{r}_x^t R^{-1} F (F^t R F)^{-1} \boldsymbol{f}_x \\
& + \boldsymbol{r}_x^t R^{-1} F (F^t R F)^{-1} F^t R^{-1} \boldsymbol{r}_x] \\
s.t. \quad & r_i - \prod_{j=1}^{d} \exp(-\theta_j |x_j - x_j^{(i)}|^{p_j}) = 0 \qquad i = 1, \ldots, n \\
& x_j^L \leq x_j \leq x_j^U \qquad j = 1, \ldots, d \\
& r_i^L \leq r_i \leq r_i^U \qquad i = 1, \ldots, n.
\end{aligned}$$

In the case where $\boldsymbol{f}_x = 1$, $\hat{y}$ and s^2 are functions of $\boldsymbol{r}_x$ only, $\hat{y}$ is even linear in $\boldsymbol{r}_x$.

Both of the above problems are non-convex. In the following we will take a closer look at how Jones *et al.* in [34] deal with the constraints and the objective functions. The constraints

$$r_i - \prod_{j=1}^{d} \exp(-\theta_j |x_j - x_j^{(i)}|^{p_j}) = 0 \qquad i = 1, \ldots, n$$

can be written as

$$\ln(r_i) + \sum_{j=1}^{d} (\theta_j |x_j - x_j^{(i)}|^{p_j}) = 0 \qquad i = 1, \ldots, n.$$

These constraints are non-convex. Jones *et al.* in [34] replace them by linear underestimators. First the n equality constraints are split up into $2n$ inequality constraints

$$\ln(r_i) + \sum_{j=1}^{d} (\theta_j |x_j - x_j^{(i)}|^{p_j}) \leq 0 \qquad i = 1, \ldots, n$$

and

$$-\ln(r_i) - \sum_{j=1}^{d} (\theta_j |x_j - x_j^{(i)}|^{p_j}) \leq 0 \qquad i = 1, \ldots, n.$$

Then the term $\ln(r_i)$ for example is replaced by a linear function $a + br_i$ that underestimates it over $[r_i^L, r_i^U]$. Similar underestimators are used to replace the other terms. In the case of $\ln(r_i)$ and $-|x_j - x_j^{(i)}|^{p_j}$ the underestimators are chords and in the case of $|x_j - x_j^{(i)}|^{p_j}$ and $-\ln(r_i)$ the underestimators are tangents. The tangents are computed at the midpoint of the relevant interval for x_j or r_i. This leads to a linearly constrained problem. So due to the relaxation of the constraints we now have a convex feasible region. To minimize $-s^2$ Jones *et al.* in [34] use a convex relaxation, αBB. This is described in detail in [23]. A nonconvex relaxation is used to minimize $\hat{y}$. These values are then used to calculate upper bounds on the expected improvement in the branch and bound.

3.8 Gradients

For some functions, the gradient at a point may be readily evaluated via known analytic expressions or techniques of automatic differentiation. They provide additional information about the behaviour of the function, and can be incorporated into the model to allow a better approximation of the function. This possibility of using gradients in the Kriging framework was investigated briefly by Morris *et al.* in [50], whose work is closely followed here, up to notational changes. We consider the unknown objective function $y_0 : [0,1]^d \to \mathbb{R}$, and let Y_0 denote the corresponding Gaussian stochastic process with $\mu_0(\boldsymbol{x}) = E[Y_0(\boldsymbol{x})]$,

$$Y_0(\boldsymbol{x}) = \mu_0(\boldsymbol{x}) + \epsilon(\boldsymbol{x}),$$

where $\epsilon(\boldsymbol{x})$ is the realization of a stationary Gaussian stochastic process. Let the partial derivatives of $y_0(\boldsymbol{x})$ and $Y_0(\boldsymbol{x})$ with respect to x_i be denoted by

$$\begin{aligned} y_i(\boldsymbol{x}) &= \frac{\partial y_0(\boldsymbol{x})}{\partial x_i} \\ Y_i(\boldsymbol{x}) &= \frac{\partial Y_0(\boldsymbol{x})}{\partial x_i}, \; i = 1 \ldots d. \end{aligned}$$

Also let $\boldsymbol{a} = (a_1, \ldots, a_d)$ be a d-vector with $\sum_i a_i = k$, where $a_i, k = 0, 1$, $i = 1, \ldots, d$, and let

$$Y_0^{(a_1,\ldots,a_d)}(\boldsymbol{x}) = \frac{\partial^k Y_0(\boldsymbol{x})}{\partial x_1^{a_1} \cdots \partial x_d^{a_d}}.$$

So for $\boldsymbol{e}^{(i)}$ a unit vector with a unit entry in position i and all other entries zero,

$$Y_0^{(e^{(i)})}(\boldsymbol{x}) = Y_i(\boldsymbol{x}).$$

We shall use the analogous notation for μ_0. The mean and covariance of the derivative process are given by

$$\begin{aligned} E(Y_0^{(a_1,\ldots,a_d)}(\boldsymbol{x})) &= \mu_0^{(a_1,\ldots,a_d)}(\boldsymbol{x}) \\ \operatorname{cov}(Y_0^{(a_1,\ldots,a_d)}(\boldsymbol{x}), Y_0^{(b_1,\ldots,b_d)}(\overline{\boldsymbol{x}})) &= \sigma^2 \operatorname{cor}(Y_0^{(a_1,\ldots,a_d)}(\boldsymbol{x}), Y_0^{(b_1,\ldots,b_d)}(\overline{\boldsymbol{x}})) \end{aligned}$$

where

$$\operatorname{cor}(Y_0^{(a_1,\ldots,a_d)}(\boldsymbol{x}), Y_0^{(b_1,\ldots,b_d)}(\overline{\boldsymbol{x}})) = (-1)^{\sum b_j} \prod_{j=1}^{d} \rho_{G_j}^{(a_j+b_j)}(x_j - \overline{x}_j) \quad (3.24)$$

where ρ_{G_j} is the autocorrelation function

$$\rho_{G_j}(x) = \exp(-\theta_j |x|^2).$$

This autocorrelation function is a special case of the previously introduced autocorrelation function $\rho_{K_j}(x) = \exp(-\theta_j|x|^{p_j})$.

This approach for using derivatives and the notation can be used for higher order partial derivatives, but here we are only interested in at most the first partial derivatives of the objective function and the Gaussian stochastic process. Therefore in our case the a_j and b_j can take values of 0 and 1 only so we are only interested in the cases where $\boldsymbol{a}$ and $\boldsymbol{b}$ are unit vectors or the zero vector $\mathbf{0}$. For notational convenience we will also denote the zero vector by $\boldsymbol{e}^{(0)} = \mathbf{0}$, and we are interested in the cases where $\boldsymbol{a} = (a_1, \ldots, a_d) = \boldsymbol{e}^{(k)}$ and $\boldsymbol{b} = (b_1, \ldots, b_d) = \boldsymbol{e}^{(l)}$ for $0 \leq k, l \leq d$. Thus

$$Y_0^{(a)} = Y_0^{(e^{(k)})}(\boldsymbol{x}) = Y_k(\boldsymbol{x})$$
$$Y_0^{(b)} = Y_0^{(e^{(l)})}(\boldsymbol{x}) = Y_l(\boldsymbol{x})$$

are the partial derivatives of Y_0 with respect to x_k and x_l, respectively. As before we consider the situation where we have sampled at n initial points $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$, where this time, besides the objective function values, we also know the first partial derivatives. Let

$$\boldsymbol{y} = (y_0(\boldsymbol{x}^{(1)}), \ldots, y_0(\boldsymbol{x}^{(n)}), y_1(\boldsymbol{x}^{(1)}), \ldots, y_1(\boldsymbol{x}^{(n)}), \ldots, y_k(\boldsymbol{x}^{(1)}), \ldots, y_k(\boldsymbol{x}^{(n)}))^t$$

be the vector of known function values and first partial derivatives. We assume that the regression term in our model is a constant, so $\mu_0(\boldsymbol{x}) = \mu$. The correlation matrix R is the $((d+1)n \times (d+1)n)$-matrix computed from (3.24) at the n design points

$$R = \operatorname{cor}(Y_0^{(e^{(k)})}(\boldsymbol{x}^{(i)}), Y_0^{(e^{(l)})}(\boldsymbol{x}^{(j)})) \qquad i, j = 1, \ldots, n, \quad k, l = 0, \ldots, d. \quad (3.25)$$

The correlation vector $\boldsymbol{r}_x$ is the $(d+1)n$-vector

$$\boldsymbol{r}_x = \operatorname{cor}(Y_0(\boldsymbol{x}), Y_0^{(e^{(k)})}(\boldsymbol{x}^{(i)})) \qquad i = 1, \ldots, n, \quad k = 0, \ldots, d$$

with entries computed from (3.24). The structure becomes clearer if we look at a partitioning of the correlation matrix R,

$$R = \begin{pmatrix} R^0 & R'^t \\ R' & R'' \end{pmatrix}.$$

Here R^0 is the $n \times n$ matrix containing the correlations of the function values. Its entries are

$$R^0_{i,j} = \operatorname{cor}(Y_0(\boldsymbol{x}^{(i)}), Y_0(\boldsymbol{x}^{(j)})), \quad i, j = 1, \ldots, n,$$

and it corresponds to the part of matrix R with $k, l = 0$ in (3.25). The matrix R' is the $nd \times n$ matrix containing the correlations of the first partial derivatives with the function values. Its entries are

$$R'_{(i-1)d+k,j} = \mathrm{cor}(Y_0^{(e^{(k)})}(\boldsymbol{x}^{(i)}), Y_0(\boldsymbol{x}^{(j)})), \quad i, j = 1, \ldots, n, \quad k = 1, \ldots d,$$

it corresponds to the entries with $l = 0$ in (3.25). Similarly, R'^t contains the entries with $k = 0$ in (3.25). And R'' is the $nd \times nd$ matrix containing the correlations of the first partial derivatives with first partial derivatives

$$R''_{(i-1)d+k,(j-1)d+l} = \mathrm{cor}(Y_0^{(e^{(k)})}(\boldsymbol{x}^{(i)}), Y_0^{(e^{(l)})}(\boldsymbol{x}^{(j)})), \quad i, j = 1, \ldots, n, \quad k, l = 1, \ldots d.$$

The correlation vector $\boldsymbol{r}_x$ can be partitioned similarly into one vector $\boldsymbol{r}^0$ of length n containing correlations of function values, and a part $\boldsymbol{r}'$ of length dn containing correlations of function values with first partial derivatives

$$\boldsymbol{r}_x = \begin{pmatrix} \boldsymbol{r}^0 \\ \boldsymbol{r}' \end{pmatrix}$$

such that $\boldsymbol{r}_{x^{(i)}}$ for $i \in \{1, \ldots, n\}$ is the same as the i-th column of the correlation matrix R. The vector $\dfrac{\partial \boldsymbol{r}_x}{\partial x_l}$ is the $(d+1)n$-vector

$$\frac{\partial \boldsymbol{r}_x}{\partial x_l} = \mathrm{cor}(Y_0^{(e^{(l)})}(\boldsymbol{x}), Y_0^{(e^{(k)})}(\boldsymbol{x}^{(i)})) \qquad i = 1, \ldots, n, \quad l = 1, \ldots, d, \quad k = 0, \ldots, d$$

which contains the correlations between derivatives at a point $\boldsymbol{x}$ and function values or derivatives at sample points. For any sample point $\boldsymbol{x}^{(i)}$, $i = 1, \ldots, n$, the vector $\dfrac{\partial \boldsymbol{r}_{x^{(i)}}}{\partial x_l}$ is the same as the $(n + (i-1)d + l)$th column of the correlation matrix R. As before, where we used only function values, we can find the predictor

$$\hat{y}(\boldsymbol{x}) = \hat{\mu} + \boldsymbol{r}_x^t R^{-1}(\boldsymbol{y} - \hat{\mu}\boldsymbol{\nu})$$

where $\hat{\mu} = (\boldsymbol{\nu}^t R^{-1} \boldsymbol{\nu})^{-1} \boldsymbol{\nu}^t R^{-1} \boldsymbol{y}$ is the maximum likelihood estimate of $\boldsymbol{\mu}$, and $\boldsymbol{\nu}$ is a vector with entries 1 in the n positions corresponding to the positions of the function values in $\boldsymbol{y}$ and 0 otherwise, so

$$\boldsymbol{\nu} = (\overbrace{1, \ldots, 1}^{n}, \overbrace{0, \ldots, 0}^{dn})^t.$$

Thus the predictor interpolates the objective function values at the points $\boldsymbol{x}^{(i)}$, $i = 1, \ldots, n$,

$$\begin{aligned} \hat{y}(\boldsymbol{x}^{(i)}) &= \hat{\mu} + \boldsymbol{r}_{x^{(i)}}^t R^{-1}(\boldsymbol{y} - \hat{\mu}\boldsymbol{\nu}) \\ &= \hat{\mu} + \boldsymbol{e}^{(i)}(\boldsymbol{y} - \hat{\mu}\boldsymbol{\nu}) \\ &= y_i. \end{aligned}$$

The predictor $\hat{y}(\boldsymbol{x})$ also interpolates the first partial derivatives at the points $\boldsymbol{x}^{(i)}$, $i = 1, \ldots, n$. To see this note that $\dfrac{\partial \boldsymbol{r}^t_{x^{(i)}}}{\partial x_j}$ is the same as the $(n + (i-1)d + j)$th row of the correlation matrix R. Therefore

$$\frac{\partial \boldsymbol{r}^t_{x^{(i)}}}{\partial x_j} R^{-1} = \boldsymbol{e}^{(n+(i-1)d+j)}$$

and

$$\begin{aligned}
\frac{\partial \hat{y}}{\partial x_j}(\boldsymbol{x}^{(i)}) &= \frac{\partial \boldsymbol{r}^t_{x^{(i)}}}{\partial x_j} R^{-1}(\boldsymbol{y} - \hat{\mu}\boldsymbol{\nu}) \\
&= \boldsymbol{e}^{(n+(i-1)d+j)}(\boldsymbol{y} - \hat{\mu}\boldsymbol{\nu}) \\
&= y_{n+(i-1)d+j} \\
&= y_j(\boldsymbol{x}^{(i)}).
\end{aligned}$$

So the predictor $\hat{y}(\boldsymbol{x})$ interpolates the objective function and the first partial derivatives of the objective function at the n sample points. As before, we can use $\hat{y}(\boldsymbol{x})$ to predict the objective function value at any point $\boldsymbol{x} \in [0,1]^d$, and now we can also use it to predict first partial derivatives.

Example 3.8.1

To give an idea of what the correlation matrix R and the vectors $\boldsymbol{r}_x$ and $\boldsymbol{\nu}$ look like when using gradients, we give a small example with $d = 1$ and $n = 2$. With $d = 1$ there is only one partial derivative to be considered and

$$\begin{aligned}
\operatorname{cor}(Y_0^{(e^{(k)})}(x^{(i)}), Y_0^{(e^{(l)})}(x^{(j)})) &= \operatorname{cor}(Y_k(x^{(i)}), Y_l(x^{(j)})) \\
&= (-1)^l \rho_{G_1}^{(k+l)}(x^{(i)} - x^{(j)}).
\end{aligned}$$

The derivatives of the Gaussian correlation function in one dimension, $\rho_{G_1}(x)$, are

$$\begin{aligned}
\rho_{G_1}^{(0)}(x) &= \exp(-\theta x^2) \\
\rho_{G_1}^{(1)}(x) &= -2\theta x \exp(-\theta x^2) \\
\rho_{G_1}^{(2)}(x) &= -2\theta \exp(-\theta x^2) + (-2\theta x)^2 \exp(-\theta x^2).
\end{aligned}$$

We have two sample points $x^{(1)}, x^{(2)} \in [0,1]$. As a shorthand notation for the differences between the two sample points we use

$$\begin{aligned}
d_1 &= x^{(1)} - x^{(2)} \\
d_2 &= x^{(2)} - x^{(1)}
\end{aligned}$$

and as a shorthand notation for the exponentials of the differences we use

$$\begin{aligned}
\exp_1 &= \exp(-\theta d_1^2) \\
\exp_2 &= \exp(-\theta d_2^2).
\end{aligned}$$

The 4×4 correlation matrix R in this case is, using (3.24),

$$\begin{pmatrix} 1 & \exp_1 & 0 & 2\theta d_1 \exp_1 \\ \exp_2 & 1 & 2\theta d_2 \exp_2 & 0 \\ 0 & -2\theta d_1 \exp_1 & 2\theta & (2\theta - (2\theta d_1)^2) \exp_1 \\ -2\theta d_2 \exp_2 & 0 & (2\theta - (2\theta d_2)^2) \exp_2) & 2\theta \end{pmatrix}.$$

The upper left 2×2 submatrix is the correlation matrix of the function values at the two sample points and corresponds to the entries with $k, l = 0$ in (3.25). The lower right 2×2 submatrix is the correlation matrix of the derivative values at the two sample points and corresponds to the entries with $k, l = 1$ in (3.25). The remaining two submatrices contain the correlations between function values and derivative values and correspond to $k = 1$, $l = 0$, and $k = 0$, $l = 1$ in (3.25).

The correlation vector $\boldsymbol{r}_x$ for any point x is

$$\boldsymbol{r}_x = \begin{pmatrix} \exp(-\theta(x - x^{(1)})^2) \\ \exp(-\theta(x - x^{(2)})^2) \\ 2\theta(x - x^{(1)}) \exp(-\theta(x - x^{(1)})^2) \\ 2\theta(x - x^{(2)}) \exp(-\theta(x - x^{(2)})^2) \end{pmatrix}$$

and the vector $\boldsymbol{\nu}$ is

$$\boldsymbol{\nu} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}.$$

The two sample points $x^{(1)}$ and $x^{(2)}$ using Latin hypercubes with centered points are chosen to be $x^{(1)} = 0.25$ and $x^{(2)} = 0.75$. The function $\hat{y}(x)$ as given in (3.8) is used as a predictor for function values and first derivatives at any point $x \in [0, 1]$.

With this approach as described above gradient information can be used to make the predictor more accurate. As we have seen, the predictor interpolates the function values at the n sample points, and it interpolates the derivatives at these points. For cases where derivatives of the objective function are readily available and can be computed without too much extra computational cost, this approach seems to be worth further investigation.

Chapter 4

Generating Sample Functions

Our algorithm is based on the Kriging approach. In the Kriging approach to optimizing expensive functions it is assumed that the objective function consists of a regression term and a realization of a stationary Gaussian stochastic process. What do such realizations of Gaussian stochastic processes look like? How does the algorithm perform with such realizations as test functions? After all the algorithm should be particularly well suited for such functions. To help answer these questions and to test how well the algorithm works we generate such realizations of Gaussian stochastic processes. These stochastic processes have certain properties, i.e. known mean and covariance, which make it easier to monitor and assess the progress and success of the algorithm. In this chapter we explain how to generate realizations or sample functions of Gaussian stochastic processes which have given autocorrelation and autocovariance functions. Because of their properties these functions are useful as test functions for our program.

Recall that we used $Y : [0,1]^d \to \mathbb{R}$ with

$$Y(\boldsymbol{x}) = \sum_{j=1}^{m} \beta_j f_j(\boldsymbol{x}) + \epsilon(\boldsymbol{x})$$

as a model for the objective function, where $\epsilon(\boldsymbol{x})$ is a stationary Gaussian stochastic process with expectation $E(\epsilon(\boldsymbol{x})) = 0$ and variance $\mathrm{var}(\epsilon(\boldsymbol{x})) = \sigma^2$. An example of an autocorrelation function of $\epsilon(\boldsymbol{x})$ and $\epsilon(\overline{\boldsymbol{x}})$, $\boldsymbol{x}, \overline{\boldsymbol{x}} \in [0,1]^d$ is the Gaussian autocorrelation function

$$\rho_G(\boldsymbol{x}, \overline{\boldsymbol{x}}) = \prod_{j=1}^{d} \exp(-\theta_j (x_j - \overline{x}_j)^2) \tag{4.1}$$

where $\theta_j > 0$, $j = 1, \ldots, d$, and this is the form of autocorrelation function and autocovariance function which will be used in what follows.

A stationary stochastic process has constant variance, σ^2 say, and in this case the relationship between the covariance $\mathrm{cov}(.,.)$ and correlation $\mathrm{cor}(.,.)$ is

given by $\text{cov}(.,.) = \sigma^2 \text{cor}(.,.)$, and similarly for the autocovariance $c(.,.)$ and the autocorrelation $\rho(.,.)$ of a stationary process, $c(.,.) = \sigma^2 \rho(.,.)$.

4.1 Generating Functions using the Correlation Matrix

Let us consider the (sample) points $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$ and a correlation matrix R with the correlations between these points as entries $R_{ij} = \rho(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)})$. We can use this to generate samples from random variables $Y(\boldsymbol{x}^{(1)}), \ldots, Y(\boldsymbol{x}^{(n)})$ with covariance and correlation matrix R. This can be done by the following method: Since R is a correlation matrix, it is positive semi-definite and we can decompose it such that

$$R = MM^t.$$

This decomposition could, for example, be a Cholesky decomposition LL^t, or $L\Sigma L^t = L\Sigma^{1/2}\Sigma^{1/2}L^t$ with lower triangular and diagonal matrix L and Σ, respectively. Alternatively we could use an eigenvalue-eigenvector decomposition $V\Sigma V^t = V\Sigma^{1/2}\Sigma^{1/2}V^t$ with the columns of V being the eigenvectors of the matrix R, and Σ being a diagonal matrix with the eigenvalues of R as its entries. Now let

$$Y(\boldsymbol{x}) = MZ(\boldsymbol{x}) \tag{4.2}$$

where $Z(\boldsymbol{x}) = (Z(\boldsymbol{x}^{(1)}), \ldots, Z(\boldsymbol{x}^{(n)}))^t$ is an n-vector of independent standard normal, $N(0,1)$ variables. By the definition of $Y(\boldsymbol{x})$ as a linear combination of normal random variables, it follows that $Y(\boldsymbol{x})$ is also normal. Because $(Z(\boldsymbol{x}^{(1)}), \ldots, Z(\boldsymbol{x}^{(n)}))$ are independent standard normal, the covariance matrix of the stochastic process $Z(\boldsymbol{x})$ is the identity matrix, i.e. $\text{Cov}(Z(\boldsymbol{x}^{(i)}), Z(\boldsymbol{x}^{(j)}))_{i,j=1,\ldots,n} = I$. Now $Y(\boldsymbol{x})$ as defined in (4.2) is a stochastic process and it is easy to see that for $\boldsymbol{x}^{(i)}$, $i = 1, \ldots, n$, the expectation of $Y(\boldsymbol{x}^{(i)})$ is

$$\begin{aligned} E(Y(\boldsymbol{x}^{(i)})) &= E(MZ(\boldsymbol{x}^{(i)})) \\ &= ME(Z(\boldsymbol{x}^{(i)})) \\ &= 0, \end{aligned}$$

since $Z(\boldsymbol{x}^{(i)})$ is standard normal, $N(0,1)$. For $\boldsymbol{x}^{(i)}$, $\boldsymbol{x}^{(j)}$, $i, j = 1, \ldots, n$ the covariance matrix of $Y(\boldsymbol{x})$ is

$$\begin{aligned} \text{Cov}(Y(\boldsymbol{x}), Y(\boldsymbol{x})) &= \text{Cov}(MZ(\boldsymbol{x}), MZ(\boldsymbol{x})) \\ &= E(MZ(\boldsymbol{x})(MZ(\boldsymbol{x}))^t) \end{aligned}$$

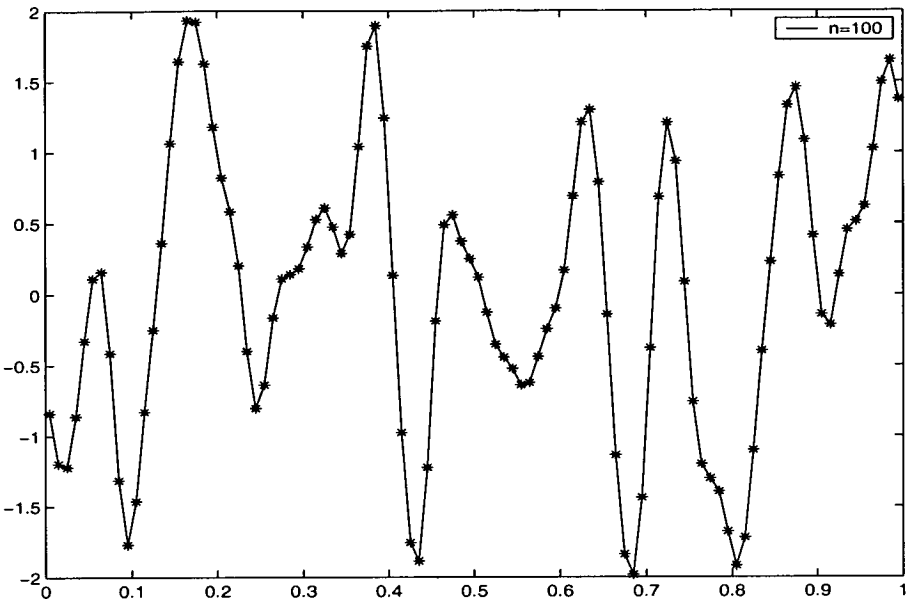

Figure 4.1: Function generated from exponential autocorrelation function with $n = 100$ and $\theta = 1000$.

$$
\begin{aligned}
&= ME(Z(\boldsymbol{x})(Z(\boldsymbol{x}))^t)M^t \\
&= M\mathrm{Cov}(Z(\boldsymbol{x}), Z(\boldsymbol{x})M^t \\
&= MIM^t \\
&= R.
\end{aligned}
$$

Therefore $Y(\boldsymbol{x})$ is a stochastic process with identical covariance and correlation matrices Cov $= R$ and process variance $\sigma^2 = 1$. Note that this approach can be used for any covariance matrix or correlation matrix, since covariance and correlation matrices are always positive semi-definite. Figure 4.1 shows an outcome using an autocorrelation function as given in (4.1) with $d = 1$, $n = 100$ and $\theta = 1000$. The 100 points are evenly spaced points $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$ in the interval $[0, 1]$. We used an eigenvector-eigenvalue decomposition for the 100×100 correlation matrix $R = V\Sigma V^t$ and set $Y(\boldsymbol{x}) = V\Sigma^{1/2}Z(\boldsymbol{x})$, so that $Y(\boldsymbol{x})$ is a linear combination of the eigenvectors of R. The entries of $Y(\boldsymbol{x})$ are plotted as $*$ against $\boldsymbol{x}$. Figure 4.2 shows the first six eigenvectors of R, i.e. the first six columns of the matrix V. These plots show functions interpolating the 100 components of the eigenvectors in the interval $[0, 1]^d$. Unfortunately this outcome is not given in function form and we cannot evaluate it at points other than $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$. The number of frequencies occurring in the eigenvectors is limited by n, the number of points used. Therefore large n should lead to a better, more general result with more frequencies used, but it leads to difficulties in the matrix decomposition of the correlation matrix, especially if correlations are high. On the other hand, the

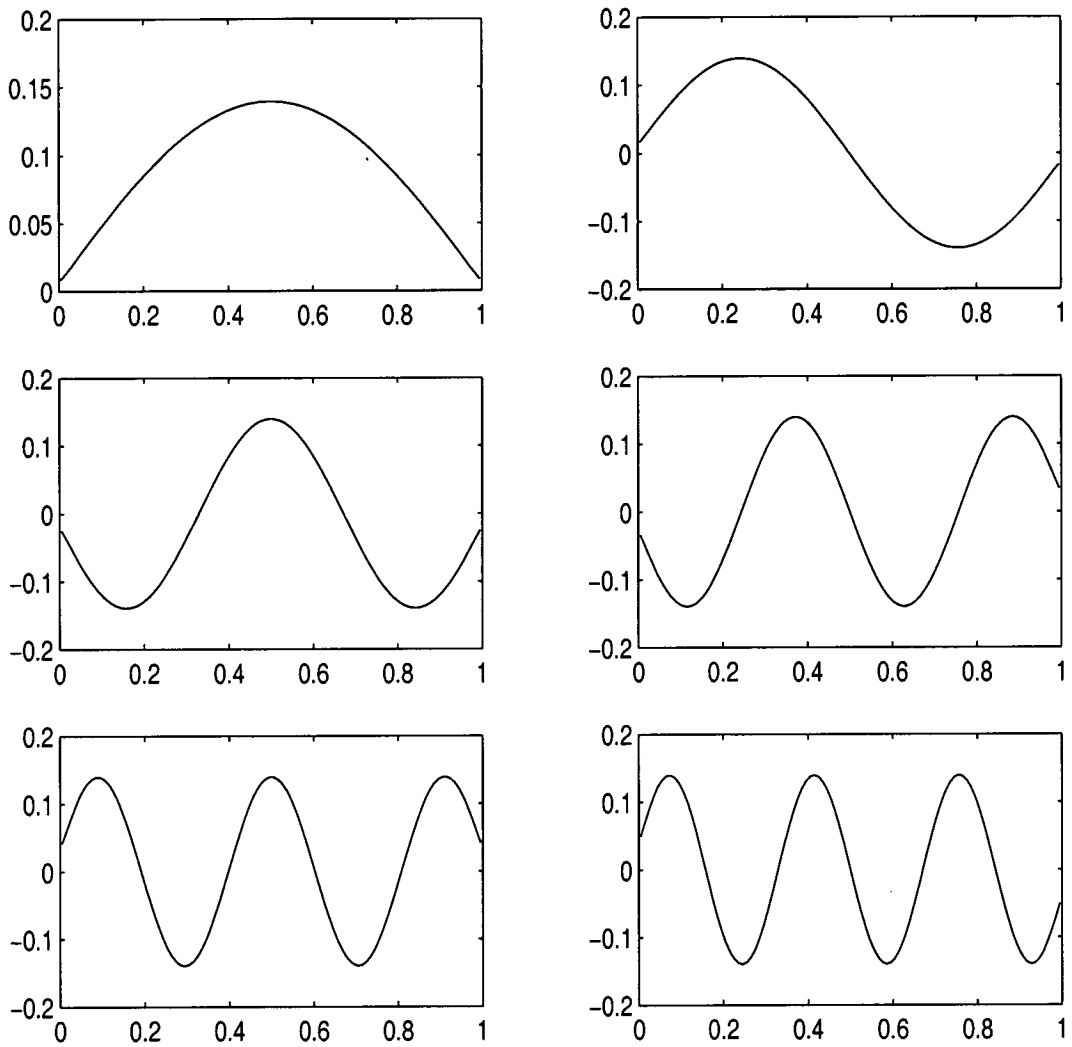

Figure 4.2: Eigenvectors of R, corresponding to the 6 largest eigenvalues

limitation on the number n of points depending on the correlation between the points seems to be largely self-correcting. High correlation means slowly changing function values, low correlation means rapidly changing function values, and to characterize a function whose function values do not display a lot of variation requires fewer points than to characterize a rapidly oscillating function. Thus only being able to use a small number of points n when correlations are high may still allow the function to be well specified. However, there remains the problem of not being able to evaluate the resulting functions at new points. This problem could be resolved if it were possible to find a closed form expression for the eigenvectors, and then the resulting function could be written as a linear combination of these. This problem is as yet unresolved and, in practice, we use another way of generating realizations of the stochastic process, which allows us to evaluate the resulting function at new points. This will be described in the rest of this chapter. Most of the background needed on stochastic processes was introduced in Section 3.1. More details can be found, for example, in Grimmett and Stirzaker's book [27].

4.2 Generating Functions Using the Autocovariance Function

Let us recall that a typical random or stochastic process is a family of random variables $\{Y(\boldsymbol{x}) : \boldsymbol{x} \in X\}$ for some set X. We are interested in the following theorem, which can also be found in [27].

Theorem 4.2.1
Given a function $c : \mathbb{R}^d \to \mathbb{R}$ such that, for all $\boldsymbol{x}, \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)} \in \mathbb{R}^d$, $z_1, \ldots, z_n \in \mathbb{R}$,

(a) $c(-\boldsymbol{x}) = c(\boldsymbol{x})$ *(symmetry)*

(b) $\sum_{j,k} c(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(j)}) z_j z_k \geq 0$, *(non-negative definiteness)*

there exists a strongly stationary process $Y(\boldsymbol{x})$ with autocovariance function c.

Our aim is now to generate samples from a Gaussian stochastic process, also in higher dimensions, that has a given autocovariance or autocorrelation function with properties as specified in Theorem 4.2.1. Note that every strongly stationary process is weakly stationary, and the covariance cov(.,.) and the autocovariance $c(.,.)$ of a weakly stationary stochastic process is invariant under shifts and only depends on distances between points. Therefore

$$c(\mathbf{0}, \boldsymbol{h}) \;=\; \mathrm{cov}(Y(\mathbf{0}), \mathbf{Y}(\boldsymbol{h}))$$

$$
\begin{aligned}
&= \operatorname{cov}(Y(\boldsymbol{x}), Y(\boldsymbol{x}+\boldsymbol{h})) \\
&= c(\boldsymbol{x}, \boldsymbol{x}+\boldsymbol{h})
\end{aligned}
$$

for all $\boldsymbol{x}, \boldsymbol{h}$, and we define

$$c(\boldsymbol{h}) = c(\boldsymbol{0}, \boldsymbol{h}).$$

Further, a weakly stationary stochastic process has constant variance σ^2, and we can define the autocorrelation function $\rho(\boldsymbol{x})$ in terms of the shift $\boldsymbol{h}$ by

$$\rho(\boldsymbol{h}) = \frac{c(\boldsymbol{h})}{\sigma^2}.$$

From the theorem above, given a positive definite symmetric function, we know there exists a stochastic process with that function as autocovariance function. We will now discuss how to generate realizations of such a stochastic process, given an autocorrelation or autocovariance function.

4.2.1 One Dimension

First, we describe how to generate functions in one dimension, which map $[0, 1] \rightarrow \mathbb{R}$. Let $c(x)$ be a non-negative definite, symmetric function. Let

$$Y(x) = \frac{1}{\sqrt{\pi}} \int_0^\infty \sqrt{\tilde{c}(\omega)}[N_1(\omega)\cos(x\omega) + N_2(\omega)\sin(x\omega)]d\omega \tag{4.3}$$

where $\tilde{c}(\omega)$ is the Fourier transform of $c(x)$, i.e.

$$\tilde{c}(\omega) = \int_{-\infty}^{\infty} c(x)\exp(-i\omega x)dx$$

and where $N_1(\omega)$ and $N_2(\omega)$ are independent normal processes. The function $c(.)$ is symmetric, and therefore its Fourier transform is its cosine transform, i.e.

$$
\begin{aligned}
\tilde{c}(\omega) &= \int_{-\infty}^{\infty} c(x)\exp(-i\omega x)dx \\
&= \int_{-\infty}^{\infty} c(x)\cos(\omega x)dx.
\end{aligned}
$$

Then $\{Y(x), x \in X\}$ is a stationary Gaussian stochastic process with expectation

$$E(Y(x)) = 0$$

and the covariance of $Y(v)$ and $Y(x)$, $v, x \in X$, is

$$\operatorname{cov}(Y(v), Y(x)) = E(Y(v)Y(x))$$

$$
\begin{aligned}
&= \frac{1}{\pi} E\left(\int_0^\infty \sqrt{\tilde{c}(\omega)}[\cos(v\omega)N_1(\omega) + \sin(v\omega)N_2(\omega)]d\omega \right. \\
&\qquad \left. \int_0^\infty \sqrt{\tilde{c}(\overline{\omega})}[\cos(x\overline{\omega})N_1(\overline{\omega}) + \sin(x\overline{\omega})N_2(\overline{\omega})]d\overline{\omega} \right) \\
&= \frac{1}{\pi} \int_0^\infty \int_0^\infty \sqrt{\tilde{c}(\omega)}\sqrt{\tilde{c}(\overline{\omega})} E\left[(\cos(v\omega)N_1(\omega) + \sin(v\omega)N_2(\omega)) \right. \\
&\qquad \left. (\cos(x\overline{\omega})N_1(\overline{\omega}) + \sin(x\overline{\omega})N_2(\overline{\omega})) \right] d\omega d\overline{\omega} \\
&= \frac{1}{\pi} \int_0^\infty \int_0^\infty \sqrt{\tilde{c}(\omega)}\sqrt{\tilde{c}(\overline{\omega})} \left(\cos(v\omega)\cos(x\overline{\omega})E[N_1(\omega)N_1(\overline{\omega})] \right. \\
&\qquad + \sin(v\omega)\cos(x\overline{\omega})E[N_2(\omega)N_1(\overline{\omega})] \\
&\qquad + \cos(v\omega)\sin(x\overline{\omega})E[N_1(\omega)N_2(\overline{\omega})] \\
&\qquad \left. + \sin(v\omega)\sin(x\overline{\omega})E[N_2(\omega)N_2(\overline{\omega})] \right) d\omega d\overline{\omega} \\
&= \frac{1}{\pi} \int_0^\infty \tilde{c}(\omega)(\cos(v\omega)\cos(x\omega) + \sin(v\omega)\sin(x\omega))d\omega \\
&= \frac{1}{2\pi} \int_{-\infty}^\infty \tilde{c}(\omega)\cos((v-x)\omega)d\omega \qquad (4.4) \\
&= c(v-x).
\end{aligned}
$$

This relies on the fact that, since N_1 and N_2 are independent with normal distribution,

$$
\begin{aligned}
E[N_1(\omega)N_1(\overline{\omega})] &= E[N_2(\omega)N_2(\overline{\omega})] \\
&= \delta(\overline{\omega} - \omega),
\end{aligned}
$$

where $\delta(.)$ is the Dirac δ function, and

$$
\begin{aligned}
E[N_2(\omega)N_1(\overline{\omega})] &= E[N_1(\omega)N_2(\overline{\omega})] \\
&= 0,
\end{aligned}
$$

and since $c(.)$ is an even function (4.4) is the inverse cosine transform of $c(.)$. For a definition and properties of the Dirac δ function see for example [27] or [59]. Therefore $Y(x)$ has the autocovariance function $c(v-x) = \mathrm{cov}(Y(v), Y(x))$ and is stationary. The above expression (4.3) for $Y(x)$ requires the calculation of two normal deviates and the evaluation of a sine and cosine.

A common way of calculating normal deviates is to use the *Box-Muller* method. This method generates pairs of independent normal random variables with joint distribution

$$
f(v,z) = \frac{1}{2\pi} \exp\left(-\frac{v^2 + z^2}{2} \right). \qquad (4.5)
$$

Details of the *Box-Muller* method as used here can be found in [16] for example. Let $U(\omega)$ and $V(\omega)$ be independent and uniformly distributed on $[0,1]$, and let

$$L(\omega) = \sqrt{-2\ln(U(\omega))}.$$

We consider the following transformation from $U(\omega)$ and $V(\omega)$ to $N_1(\omega)$ and $N_2(\omega)$,

$$\begin{aligned} N_1(\omega) &= L(\omega)\cos(V(\omega)2\pi) \\ N_2(\omega) &= -L(\omega)\sin(V(\omega)2\pi). \end{aligned}$$

The random variables $N_1(\omega)$ and $N_2(\omega)$ can be shown to have joint distribution function $f(.,.)$ as in (4.5), and are independently normally distributed. From this it follows that

$$\begin{aligned} N_1(\omega)\cos(x\omega) + N_2(\omega)\sin(x\omega) &= L(\omega)\cos(V(\omega)2\pi)\cos(x\omega) \\ &\quad -L(\omega)\sin(V(\omega)2\pi)\sin(x\omega) \\ &= L(\omega)\cos(x\omega + V(\omega)2\pi) \end{aligned}$$

and so (4.3) can be written in the following form

$$Y(x) = \frac{1}{\sqrt{\pi}}\int_0^\infty \sqrt{\tilde{c}(\omega)}L(\omega)\cos(x\omega + V(\omega)2\pi)d\omega. \tag{4.6}$$

Therefore, for the random variables $L(\omega)$ and $V(\omega)$ as above, this is a Gaussian stochastic process. The form (4.6) avoids the evaluation of the sine, and is more efficient than (4.3).

For realizations (specific outcomes) of $L(\omega)$ and $V(\omega)$, $Y(x)$ is a realization of such a process and a function of x only. In practice we use a discrete approximation to the integral in (4.6), and we want the autocovariance $c_D(.)$ of this discrete approximation to be a good approximation to $c(.)$. Generally, for a discrete approximation of $Y(x)$ as in (4.6) we can use points $\omega_n \in \mathbb{R}^+$ and let

$$Y(x) = \frac{1}{\sqrt{\pi}}\sum_{n=0}^{\infty} \sqrt{a_n}\sqrt{\tilde{c}(\omega_n)}L_n\cos(x\omega_n + V_n 2\pi), \tag{4.7}$$

where every point ω_n has an interval size or area a_n associated with it. The points ω_n do not have to be equally spaced. Indeed there is an argument for not spacing them equally, but increasing their distances where the autocovariance function $c(.)$ varies less, for example if $c(.)$ is an exponential and dies down rapidly. Then larger values of a_n could be taken for higher values of ω. But we can of course use equally spaced points $\omega \in \mathbb{R}^+$ with $\omega = nh$, in which case all points nh for $n \geq 1$

have an interval of length h associated with them, and the point with $n = 0$ has an interval of length $h/2$ associated with it. In this case (4.7) becomes

$$Y(x) = \sqrt{\frac{h}{\pi}}\left[\frac{\sqrt{\tilde{c}(0)}L_0}{\sqrt{2}} + \sum_{n=1}^{\infty}\sqrt{\tilde{c}(nh)}L_n\cos(xnh + V_n 2\pi)\right]. \tag{4.8}$$

From (4.8) and by analogy with (4.4) we can see that the the autocovariance function $c_D(.)$ of the stochastic process $\{Y(x), x \in X\}$ as in (4.8) is

$$\begin{aligned} c_D(v-x) &= \operatorname{cov}(Y(v), Y(x)) \qquad (4.9)\\ &= \frac{h}{\pi}\left(\frac{\tilde{c}(0)}{2} + \sum_{n=1}^{\infty}\tilde{c}(nh)\cos((v-x)nh)\right)\\ &= \frac{h}{2\pi}\sum_{n=-\infty}^{\infty}\tilde{c}(nh)\cos((v-x)nh)\\ &= \frac{h}{2\pi}\sum_{n=-\infty}^{\infty}\tilde{c}(nh)\exp(i(v-x)nh). \end{aligned}$$

Using the Poisson summation formula, for this see for example [20],we can rewrite (4.9) as

$$c_D(v-x) = \sum_{n=-\infty}^{\infty} c\left(\frac{2\pi n}{h} + v - x\right). \tag{4.10}$$

What is a good choice for the interval size h here to achieve a good approximation to the autocovariance function $c(.)$? The error of the autocovariance function $c_D(.)$ of the process generated by (4.8) compared with the autocovariance function $c(.)$ is

$$\begin{aligned} c_D(v-x) - c(v-x) &= \sum_{n=-\infty}^{\infty} c\left(\frac{2\pi n}{h} + v - x\right) - c(v-x) \qquad (4.11)\\ &= \sum_{n\neq 0} c\left(\frac{2\pi n}{h} + v - x\right), \qquad (4.12) \end{aligned}$$

and this sum has to be small to achieve a good approximation to $c(.)$. For the case of the exponential autocovariance function $c(x) = \exp(-x^2)$ the error term is

$$c_D(v-x) - c(v-x) = \sum_{n\neq 0}\exp\left(-\left(\frac{2\pi n}{h} + v - x\right)^2\right), \tag{4.13}$$

and we should choose h to make this small. If v and x are in an interval of length 1 the sum in (4.13) is dominated by the term $\exp\left(-\left(\frac{2\pi}{h} - 1\right)^2\right)$ and we should choose h to be small compared with 2π so that $\frac{2\pi}{h} - 1 > 0$ and is large

enough to make this leading term small. For example a choice of $h = 1$ gives $\exp\left(-(2\pi - 1)^2\right) \approx 7.55 \times 10^{-13}$. Now we can also fix a maximum leading error term and find the maximum stepsize h that is needed to make the actual leading error term smaller than this fixed maximum error. For example if we want

$$\exp\left(-\left(\frac{2\pi}{h} - 1\right)^2\right) \leq \exp(-36)$$

then we have to choose h such that

$$h \leq \frac{2\pi}{7}. \tag{4.14}$$

In practice we use a finite sum to approximate (4.8): if N is large enough and h is small enough then

$$Y(x) = \sqrt{\frac{h}{\pi}}\left[\frac{\sqrt{\tilde{c}(0)}L_0}{\sqrt{2}} + \sum_{n=1}^{N}\sqrt{\tilde{c}(nh)}L_n\cos(xnh + V_n 2\pi)\right] \tag{4.15}$$

should give a good approximation to a realization of a Gaussian stochastic process with autocovariance function $c(x)$.

This method of generating functions with autocovariance function $c(x)$ can be generalized to the case where a scaling parameter θ is introduced into the autocovariance function, as in the autocorrelation function introduced in Chapter 3,

$$c_G = \prod_{j=1}^{d}\exp\left(-\theta_j x_j^2\right). \tag{4.16}$$

Processes with this autocorrelation or autocovariance function can be generated by scaling $Y(x)$ where $Y(x)$ is generated as described above. The scaling can be done as follows: define

$$Y_{\sqrt{\theta}}(x) \;=\; Y(\sqrt{\theta}x).$$

For $Y(x)$ generated by

$$Y(x) = \frac{1}{\sqrt{\pi}}\int_0^\infty \sqrt{\tilde{c}(\omega)}L(\omega)\cos(x\omega + V(\omega)2\pi)d\omega \tag{4.17}$$

the covariance of $Y_{\sqrt{\theta}}(v)$ and $Y_{\sqrt{\theta}}(x)$ is

$$\begin{aligned}\operatorname{cov}(Y_{\sqrt{\theta}}(v), Y_{\sqrt{\theta}}(x)) &= E(Y_{\sqrt{\theta}}(v), Y_{\sqrt{\theta}}(x)) \\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty}\tilde{c}(\omega)\cos((\sqrt{\theta}v - \sqrt{\theta}x)\omega)d\omega \\ &= c(\sqrt{\theta}(v - x)). \end{aligned} \tag{4.18}$$

For $c(x) = \exp(-x^2)$ this is

$$\operatorname{cov}(Y_{\sqrt{\theta}}(v), Y_{\sqrt{\theta}}(x)) = \exp\left(-\theta(v-x)^2\right),$$

as required.

How does applying such a scaling affect the stepsize or number of frequencies we should use in the discretization? For large θ more frequencies and a smaller stepsize will be needed to achieve a good approximation to the autocovariance function. Therefore we return to the question of what can be used as a guideline for the choice of the stepsize in a regular discretization to achieve a good approximation to the autocovariance function $c_{\sqrt{\theta}}$, and consider the example of the Gaussian autocovariance function as in (4.16). An analogous derivation to (4.9) gives

$$\begin{aligned} c_{D\sqrt{\theta}}(v-x) &= \operatorname{cov}(Y_{\sqrt{\theta}}(v), Y_{\sqrt{\theta}}(x)) \\ &= \frac{h}{2\pi} \sum_{n=-\infty}^{\infty} \tilde{c}(nh) \cos(\sqrt{\theta}(v-x)nh) \\ &= \sum_{n=-\infty}^{\infty} c\left(\frac{2\pi n}{h} + \sqrt{\theta}(v-x)\right) \end{aligned}$$

and similarly to before, the leading error term for the exponential autocorrelation or autocovariance function is

$$\exp\left(-\left(\frac{2\pi}{h} - \sqrt{\theta}\right)^2\right),$$

and h should be chosen small compared with $\dfrac{2\pi n}{\sqrt{\theta}}$ to make this small.

As a guideline for the 1-dimensional case we choose the smallest number of frequencies n_ω which makes the maximum leading error term $\leq \exp(-36)$, i.e.

$$\exp\left(-\left(\frac{2\pi}{h} - \sqrt{\theta}\right)^2\right) \leq \exp(-36).$$

With a fixed upper bound ω_{ub} for the frequencies, the stepsize h and the number of frequencies n_ω are related by

$$h = \frac{\omega_{ub}}{n_\omega - 1},$$

and by (4.19), similarly to (4.14), h should be no larger than $\dfrac{2\pi}{6+\sqrt{\theta}}$. So we set

$$\frac{2\pi}{6+\sqrt{\theta}} \leq \frac{\omega_{ub}}{n_\omega - 1}$$

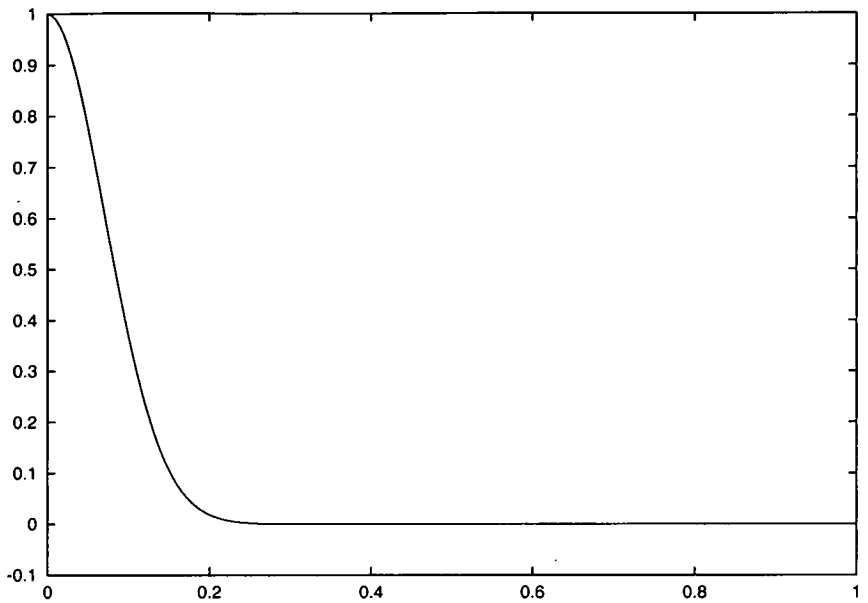

Figure 4.3: Autocovariance function c_D of process generated with $n_\omega = 27$ and $\omega_{ub} = 10$

and use as a guideline for the number of frequencies n_ω,

$$n_\omega \geq \left\lceil \frac{\omega_{ub}\left(6+\sqrt{\theta}\right)}{2\pi} \right\rceil + 1.$$

Figure 4.3 shows an example of an autocovariance function $c_D(x)$ of a process generated with $c(x) = \exp(-100x^2)$, $n_\omega = 27$, and $\omega_{ub} = 10$. Enough frequencies and a high enough upper bound on the frequencies are used, and the function $c_D(x)$ is a good match to the function $c(x) = \exp(-100x^2)$. Figure 4.4 shows an autocovariance function of a process generated with too few frequencies. With $n_\omega = 11$ used to generate the process, the autocovariance function $c_D(x)$ is periodic with period $\frac{2\pi}{h\sqrt{\theta}} \approx 0.628$. An example where a too small upper bound $\omega_{ub} = 2$ was used is shown in Figure 4.5.

4.2.2 Two Dimensions

Essentially the same approach as described above can be used to generate functions of more variables. We use $||\boldsymbol{\omega}||$ to denote the Euclidean or 2-norm of $\boldsymbol{\omega}$ such that

$$||\boldsymbol{\omega}|| = \left(\sum_{j=1}^{d} \omega_j^2\right)^{1/2}.$$

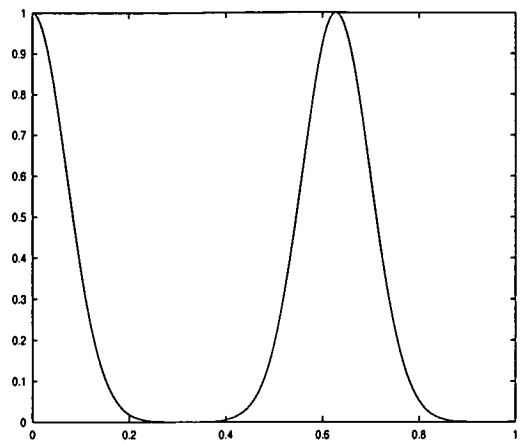

Figure 4.4: Autocovariance function c_D of process generated with $n_\omega = 11$ and $\omega_{ub} = 10$

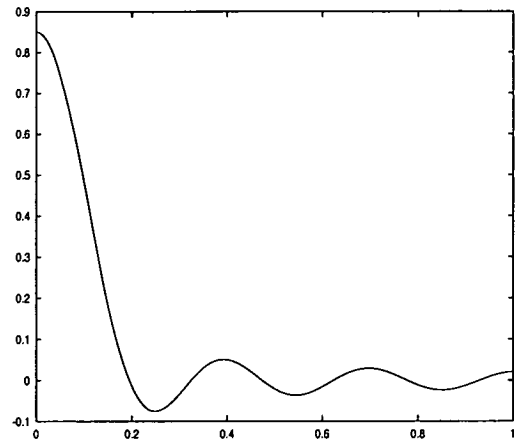

Figure 4.5: Autocovariance function c_D of process generated with $n_\omega = 27$ and $\omega_{ub} = 2$

In higher dimensions the autocovariance function $c_d(\boldsymbol{x})$ with $\boldsymbol{x} = (x_1, \ldots, x_d)^t$ has Fourier transform

$$\tilde{c}_d(\boldsymbol{\omega}) = \int_{\mathbb{R}^d} \exp(-i(\boldsymbol{x}^t\boldsymbol{\omega}))c_d(\boldsymbol{x})d\boldsymbol{x}.$$

For the case when $d = 2$ we can obtain $\tilde{c}_2$ as follows. Let

$$\begin{aligned} s_1 &= \frac{\omega_1 x_1 + \omega_2 x_2}{\|\boldsymbol{\omega}\|} \\ s_2 &= \frac{-\omega_2 x_1 + \omega_1 x_2}{\|\boldsymbol{\omega}\|} \end{aligned}$$

so

$$\begin{aligned} \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} &= \frac{1}{\|\boldsymbol{\omega}\|} \begin{pmatrix} \omega_1 & \omega_2 \\ -\omega_2 & \omega_1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ &= \begin{pmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \end{aligned}$$

where

$$\begin{aligned} \cos\phi &= \frac{\omega_1}{\|\boldsymbol{\omega}\|} \\ \sin\phi &= \frac{\omega_2}{\|\boldsymbol{\omega}\|} \end{aligned}$$

and integrate over s_1 and s_2. The determinant of the Jacobian of the transformation is 1, therefore

$$\begin{aligned} \tilde{c}_2(\boldsymbol{\omega}) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \exp(-i\|\boldsymbol{\omega}\|s_1)c_2^{\phi}(s_1, s_2)ds_2ds_1 \\ &= \int_{-\infty}^{\infty} \exp(-i\|\boldsymbol{\omega}\|s_1) \int_{-\infty}^{\infty} c_2^{\phi}(s_1, s_2)ds_2ds_1, \end{aligned}$$

where

$$c_2^{\phi}(s_1, s_2) = c_2(s_1 \cos\phi - s_2 \sin\phi, s_1 \sin\phi + s_2 \cos\phi).$$

Let

$$c_{1D}^{\phi}(s_1) = \int_{-\infty}^{\infty} c_2^{\phi}(s_1, s_2) ds_2 \tag{4.19}$$

then

$$\begin{aligned} \tilde{c}_2(\boldsymbol{\omega}) &= \int_{-\infty}^{\infty} \exp(-i\|\boldsymbol{\omega}\| s_1) c_{1D}^{\phi}(s_1) ds_1 && (4.20) \\ &= \tilde{c}_{1D}^{\phi}(\|\boldsymbol{\omega}\|). && (4.21) \end{aligned}$$

Note that if the autocovariance function $c_2(\boldsymbol{x})$ is circularly symmetric then the above transformation is independent of the angle of rotation, and c_{1D}^{ϕ} is independent of ϕ, and depends only on the distance of the point $\boldsymbol{\omega}$ from the origin. This result holds for general d. We can use $\tilde{c}_2(\boldsymbol{\omega})$ to generate a realization of a 2-dimensional Gaussian stochastic process as follows,

$$\begin{aligned} Y(\boldsymbol{x}) &= \frac{1}{2\pi} \int_{\mathbb{R}^2} \sqrt{\tilde{c}_2(\boldsymbol{\omega})} L(\boldsymbol{\omega}) \\ &\qquad \cos((\boldsymbol{x}^t \boldsymbol{\omega}) + V(\boldsymbol{\omega}) 2\pi) d\boldsymbol{\omega}, && (4.22) \end{aligned}$$

such that

$$\begin{aligned} \operatorname{cov}(Y(x), Y(v)) &= E(Y(x)Y(v)) \\ &= \frac{1}{4\pi^2} \int_{\mathbb{R}^2} \sqrt{\tilde{c}_2(\boldsymbol{\omega})} \cos(\boldsymbol{x} - \boldsymbol{v})^t \boldsymbol{\omega} d\boldsymbol{\omega} \\ &= c_2(\boldsymbol{x} - \boldsymbol{v}). \end{aligned}$$

As mentioned before, a possible choice for the autocovariance function in one dimension is $c(x) = \exp(-x^2)$. This has Fourier transform

$$\tilde{c}(\omega) = \sqrt{\pi} \exp\left(-\frac{\omega^2}{4}\right).$$

In the 2-dimensional case the corresponding circularly symmetric autocovariance function is $c_2(x_1, x_2) = \exp(-(x_1^2 + x_2^2))$ and from (4.19) it follows that

$$\begin{aligned} c_{1D}(s_1) &= \int_{-\infty}^{\infty} \exp(-s_1^2 - s_2^2) ds_2 \\ &= \sqrt{\pi} \exp(-s_1^2) \end{aligned}$$

and from (4.20),

$$\begin{aligned} \tilde{c}_2(\omega_1, \omega_2) &= \sqrt{\pi} \tilde{c}_{1D}(\|\boldsymbol{\omega}\|) \\ &= \pi \exp\left(-\frac{\|\boldsymbol{\omega}\|^2}{4}\right). \end{aligned}$$

For a discrete approximation of $Y(\boldsymbol{x})$ we can use points $\boldsymbol{\omega}^{(k)}$ on a grid and split the area of integration into small areas surrounding the points $\boldsymbol{\omega}^{(k)}$. Let N_ω denote the total number of points, then (4.22) can be approximated by

$$Y(\boldsymbol{x}) = \frac{1}{2\pi}\left[\sum_{k=1}^{N_\omega} \sqrt{a_k}\sqrt{\tilde{c}_2(\boldsymbol{\omega})} L_k \cos(\boldsymbol{x}^t\boldsymbol{\omega}^{(k)} + V_k 2\pi)\right] \tag{4.23}$$

where a_k denotes the area of volume associated with the point $\boldsymbol{\omega}^{(k)}$, for purposes of approximating the integral.

We now consider alternative discretizations (grids) for functions of two variables, i.e. $d = 2$. One possible choice of grid is a rectangular grid with points (nh, mh), h an interval size or stepsize and $n, m \in \mathbb{Z}$, such that

$$Y(x_1, x_2) = \frac{h}{2\pi}\sum_{n=-N}^{N}\sum_{m=-N}^{N} \sqrt{\tilde{c}_2(nh, mh)} L_{nm} \cos((x_1 nh + x_2 mh) + V_{nm} 2\pi)$$

where $a_k = h^2$ is the area associated with each point (nh, mh) for $n \neq 0$. The two points (nh, mh) and $(-nh, mh)$ have the same contribution to the function $Y(x_1, x_2)$ and we can restrict ourselves to one half of the grid used above, which introduces a factor of $\sqrt{2}$, and $Y(x_1, x_2)$ becomes

$$\begin{aligned} Y(x_1, x_2) \doteq \frac{h}{2\pi}\Bigg[&\sum_{m=-N}^{N} \sqrt{\tilde{c}_2(0, mh)} L_m \cos(x_2 mh + V_m 2\pi) \\ &+\sqrt{2}\sum_{n=1}^{N}\sum_{m=-N}^{N} \sqrt{\tilde{c}_2(nh, mh)} L_{nm} \cos((x_1 nh + x_2 mh) + V_{nm} 2\pi)\Bigg]. \end{aligned}$$

In practice we leave out the points for which $\|\boldsymbol{\omega}\| > \omega_{ub}$. With ω_{ub} large enough and the autocovariance function we are using, $\tilde{c}_{1D}^{\phi}(\|\boldsymbol{\omega}\|)$ is small for $\|\boldsymbol{\omega}\| > \omega_{ub}$ and the contribution of the points $\boldsymbol{\omega}$ with $\|\boldsymbol{\omega}\| > \omega_{ub}$ to the function generated is negligible. Figure 4.6 shows an example of a regular rectangular grid with 11×21 squares and 169 points for which $\|\boldsymbol{\omega}\| \leq 10$ (indicated by the dashed half circle). If the discretization used is fine enough, then all discretization methods will produce an accurate approximation to the continuous stochastic process. However, with a small number of grid points, different discretizations or different grid-spacing will produce different stochastic processes. For example a function generated from a uniform rectangular grid for $\omega_1 = nh$, $\omega_2 = mh$, $n, m \leq N$ can display obvious periodicity if N is not large and h not small enough. This can be avoided by choosing N to be larger and h smaller, or for example by using a circular grid as a discretization for (ω_1, ω_2). In some cases functions generated using a rectangular grid display obvious periodicity, whereas functions generated using a circular grid with almost the same number of points (ω_1, ω_2) as in the

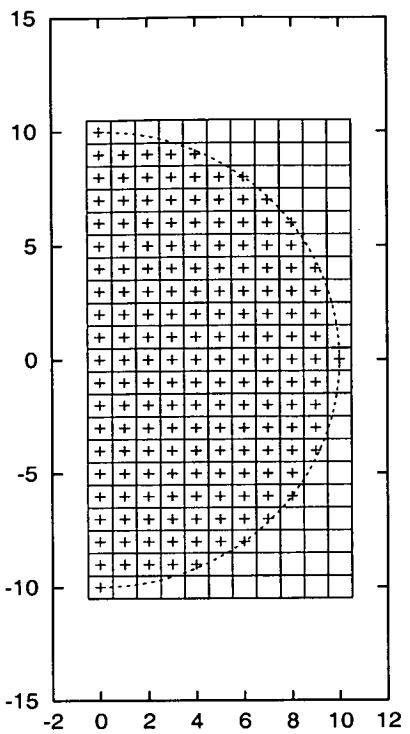

Figure 4.6: Rectangular grid with 11 intervals and 169 points

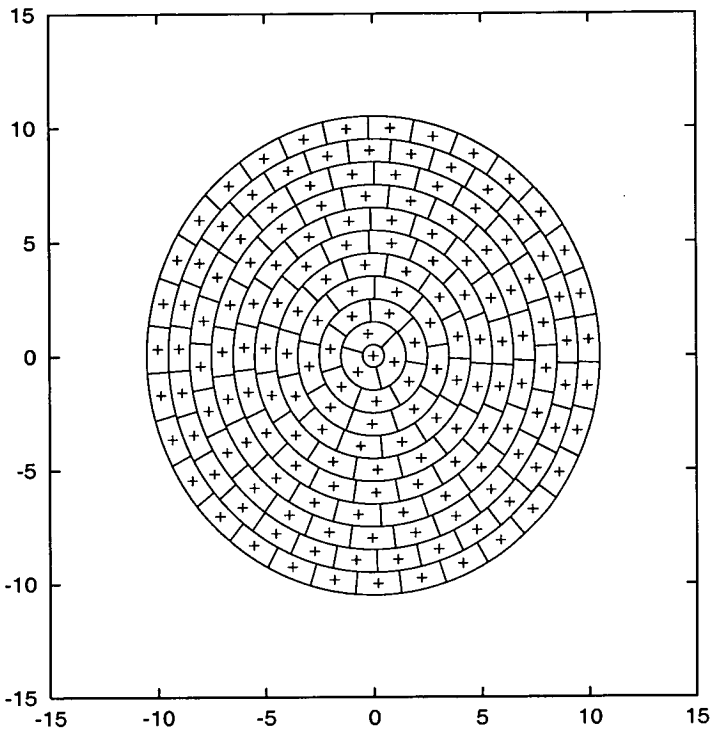

Figure 4.7: Circular grid with 11 radii and 173 points

rectangular case, display no obvious periodicity. To construct a circular grid we choose an upper bound ω_{ub} for the frequencies ω_1, ω_2, such that $\tilde{c}_2(\omega_1, \omega_2)$ is small and a number n_α of radii α_i with $0 \leq \alpha_i \leq \omega_{ub}$ for $i = 1, \ldots, n_\alpha$. We choose α_i equally spaced: $\alpha_i = (i-1)\Delta\alpha$ where $\Delta\alpha = \omega_{ub}/(n_\alpha - 1)$. On each of the n_α circles we then choose a number of equally spaced points around the circle as the points for our discretization. The number of points chosen on the circle with radius α_i is k_i, the smallest odd number which is greater than or equal to the integer part of $\pi\alpha_i/\Delta\alpha$,

$$k_i = 2\left\lfloor \frac{\pi\alpha_i/\Delta\alpha}{2} \right\rfloor + 1.$$

The reason for choosing k_i odd is to avoid two points $\boldsymbol{\omega}$ with the same radius $\|\boldsymbol{\omega}\| = \alpha_i$ and angle π apart, as these would generate the same terms in the Fourier expansion. The angles of the points on the circle are determined by $2\pi/k_i$, where the first point has a random angle from a uniform $U[0, 2\pi/k_i]$ distribution. So on the circle of radius α_i we choose the points (ω_1, ω_2) given by

$$\begin{aligned} \phi &= U\left[0, \frac{2\pi}{k_i}\right] + j\frac{2\pi}{k_i} \\ \omega_1 &= \alpha_i \cos\phi \\ \omega_2 &= \alpha_i \sin\phi \end{aligned}$$

for $j = 0, \ldots, k_i - 1$ as our discretization. Figure 4.7 shows an example of a circular grid with 11 radii and 173 points. As in the rectangular grid shown in Figure 4.6, we used points ω in this grid for which $\|\omega\| \leq 10$. The single point corresponding to $\alpha_1 = 0$ now has an area of

$$a_0 = \pi \left(\frac{1}{2}\Delta\alpha\right)^2$$

associated with it. Any other point lying on the circle of radius α_i, $1 < i \leq n_\alpha$ has area

$$\begin{aligned} a_i &= \frac{\pi\left[\left(\left(i+\frac{1}{2}\right)\Delta\alpha\right)^2 - \left(\left(i-\frac{1}{2}\right)\Delta\alpha\right)^2\right]}{k_i} \\ &= \frac{\pi 2 i \Delta\alpha^2}{k_i} \end{aligned}$$

associated with it, and the corresponding terms in the summation are scaled by the square root of the associated area, $\sqrt{a_i}$. To generate 2-dimensional sample functions from the circular grid, let

$$N_\omega = \sum_{i=1}^{n_\alpha} k_i,$$

so that N_ω is the total number of points on the circular grid, and we use (4.23) as above to generate functions $Y(x_1, x_2)$.

Similarly to the 1-dimensional case with evenly spaced points we can find an error estimate for the regular rectangular grid in two dimensions. By analogy with (4.9) and (4.11) the error is

$$\sum_{(n,m)\neq(0,0)} c_2\left(\frac{2\pi n}{h} + v_1 - x_1, \frac{2\pi m}{h} + v_2 - x_2\right)$$

and as in the 1-dimensional case h should be small compared with 2π to make the leading term in the error small. This leads to the same choice of h as in the 1-dimensional case, see (4.14).

In two dimensions, as in the 1-dimensional case, scaling factors $\boldsymbol{\theta}$ can be introduced into the autocovariance function by scaling $Y(\boldsymbol{x})$. Let $Y(\boldsymbol{x})$ be as in (4.22), generated from the function

$$c_G(\boldsymbol{x}) = \prod_{j=1}^{d} \exp\left(-x_j^2\right),$$

and let

$$Y_{\sqrt{\theta}}(x_1, x_2) = Y\left(\sqrt{\theta_1}x_1, \sqrt{\theta_2}x_2\right). \tag{4.24}$$

By analogy with (4.18), the covariance of $Y_{\sqrt{\theta}}(x_1, x_2)$ and $Y_{\sqrt{\theta}}(v_1, v_2)$ is

$$\mathrm{cov}(Y_{\sqrt{\theta}}(v_1, v_2), Y_{\sqrt{\theta}}(x_1, x_2)) = \exp\left(-\theta_1(v_1 - x_1)^2 - \theta_2(v_2 - x_2)^2\right)$$

as required.

More generally, by applying a map $A : \mathbb{R}^d \to \mathbb{R}^d$ to $\boldsymbol{x} \in \mathbb{R}^d$ and defining

$$Y_A(\boldsymbol{x}) = Y(A(\boldsymbol{x}))$$

other Gaussian processes can be generated. These processes are not necessarily stationary and there is not always a simple form for the corresponding autocovariance function. This is the case for example if $A(\boldsymbol{x})$ is a non-linear distortion of $\boldsymbol{x}$. If we apply a linear transformation $M : \mathbb{R}^d \to \mathbb{R}^d$ to $\boldsymbol{x} \in \mathbb{R}^d$ then the corresponding covariance and autocovariance function are

$$\begin{aligned} \mathrm{cov}(Y_M(\boldsymbol{v}), Y_M(\boldsymbol{x})) &= E(Y_M(\boldsymbol{v}), Y_M(\boldsymbol{x})) \\ &= E(Y(M\boldsymbol{v}), Y(M\boldsymbol{x})) \\ &= c(M(\boldsymbol{x} - \boldsymbol{v})) \\ &=: c_M((\boldsymbol{x} - \boldsymbol{v})), \end{aligned}$$

so we can generate realizations $Y_M(\boldsymbol{x})$ from a stationary Gaussian process with autocovariance function $c_M(.)$ by applying the transformation M to $\boldsymbol{x}$. Examples of transformations which could be of interest are rotations. The particular transformation mentioned above in (4.24) is a special case of a linear transformation with

$$M = \begin{pmatrix} \sqrt{\theta_1} & 0 \\ 0 & \sqrt{\theta_2} \end{pmatrix}.$$

How does applying such a scaling affect the stepsize or number of frequencies we should use in the discretization? For large $\boldsymbol{\theta}$ more frequencies and a smaller stepsize will be needed to achieve a good approximation to the autocovariance function. So we return to the question of what we can be used as a guideline for the choice of the stepsize in a regular grid in two dimensions to achieve a good approximation to the autocovariance function $c_{\sqrt{\theta}}$, and consider the example of the exponential autocovariance function. In two dimensions the error estimation and choice of stepsize works similarly to the 1-dimensional case, and the leading error term for the exponential covariance function is

$$\exp\left(-\left(\frac{2\pi}{h_1} - \sqrt{\theta_1}\right)^2 - \left(\frac{2\pi}{h_2} - \sqrt{\theta_2}\right)^2\right)$$

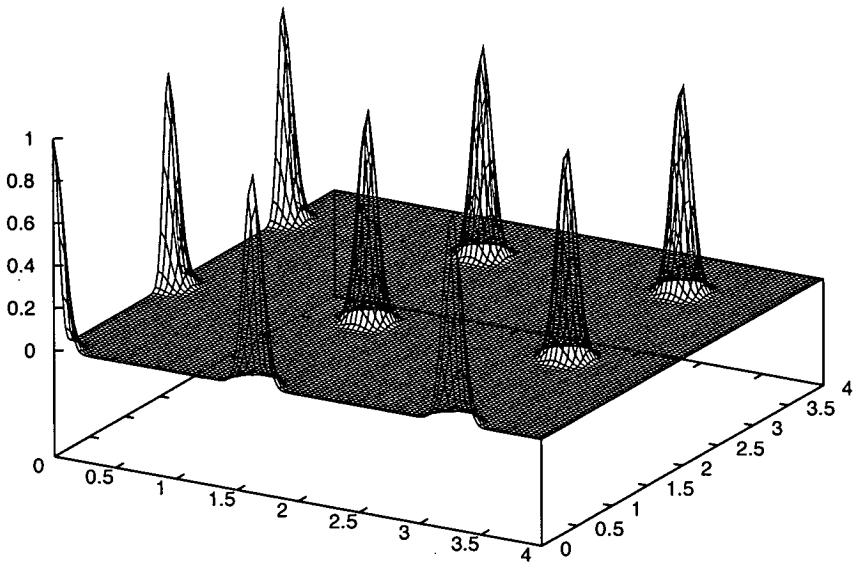

Figure 4.8: Autocovariance function of a stochastic process generated with a rectangular grid

and h_1 should be chosen small compared with $\frac{2\pi}{\sqrt{\theta_1}}$ and h_2 small compared with $\frac{2\pi}{\sqrt{\theta_2}}$ to make the leading error term small.

In two dimensions we use the same guideline for the number of frequencies and the stepsize as in one dimension. In half a rectangular grid we choose n_ω and h by the above guidelines, and similarly in the circular grid, where we use $\Delta r = h$ for the difference in the radii and n_ω as the number of different radii used. For example for $\theta = 100$ using this guideline gives

$$n_\omega \geq \left\lceil \frac{\omega_{ub}\,(6+10)}{2\pi} \right\rceil + 1$$

and with $\omega_{ub} = 10$,

$$n_\omega = 27$$

in one dimension. Figure 4.8 shows the autocovariance function of a process generated with

$$c(\boldsymbol{x}) = \prod_{i=1}^{2} \exp\left(-\theta_i x_i^2\right)$$

and $\theta_1 = \theta_2 = 100$. The number of frequencies used on the rectangular half grid is 1085, which is 27×53 minus the number of frequencies $\boldsymbol{\omega}$ with $||\boldsymbol{\omega}|| \geq 10$.

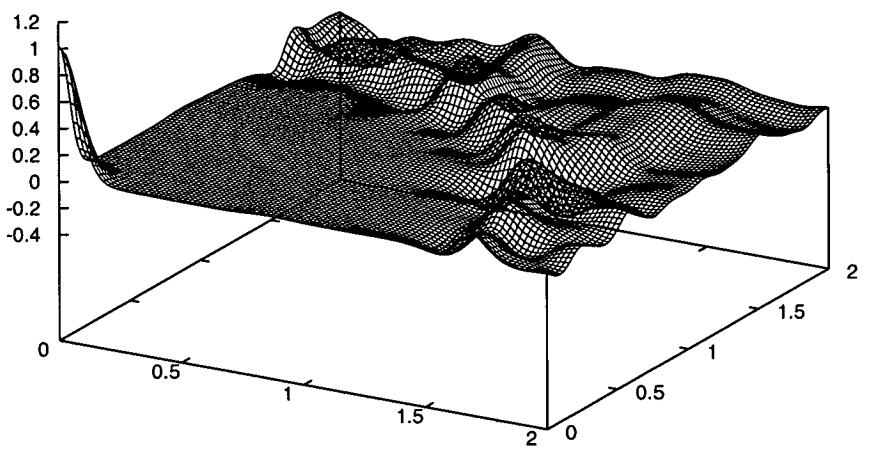

Figure 4.9: Autocovariance function of a stochastic process generated with a circular grid

Figure 4.10 shows a contour plot of this autocovariance function. While it is a good approximation to an exponential on $[0,1]^2$, it is periodic with period $\frac{2\pi}{h\sqrt{\theta}}$ and $h = \frac{\omega_{ub}}{n_\omega - 1}$, and not a good approximation to an exponential function much outside $[0,1]^2$. Figures 4.8 and 4.10 are plots of

$$c_D(x_1,x_2) = \frac{h}{2\pi}\left[\sum_{m=-N}^{N}\tilde{c}_2(0,mh)\cos(x_2 mh) + \sqrt{2}\sum_{n=1}^{N}\sum_{m=-N}^{N}\tilde{c}_2(nh,mh)\cos(x_1 nh + x_2 mh)\right]$$

which should be a good approximation to $c(\boldsymbol{x})$ for a large number of frequencies used, i.e. for large N. Similarly, Figures 4.9 and 4.11 show the autocovariance function and its contours for a process generated from a circular grid. Again, this is a good approximation to an exponential on $[0,1]^2$, but not outside this region.

Examples of sample paths for frequencies chosen from a rectangular grid and a circular grid are shown in Figures 4.12 and 4.13. In both cases the autocovariance function $c(\boldsymbol{x}) = \prod_j \exp(-\theta_j x_j^2) = \exp(-(\theta_1 x_1^2 + \theta_2 x_2^2))$ with $\theta_1 = \theta_2 = 400$ was used. Figure 4.12 shows the contours of a function which was generated using the rectangular grid, as shown in Figure 4.6. The function is obviously

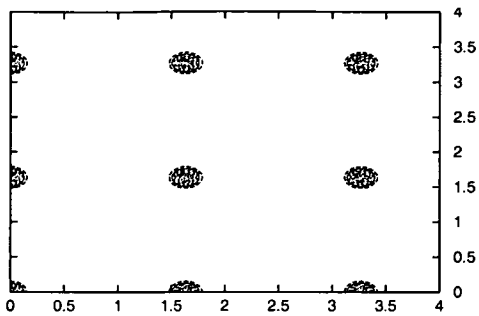

Figure 4.10: Contours of autocovariance function c_D for a process generated from a rectangular grid

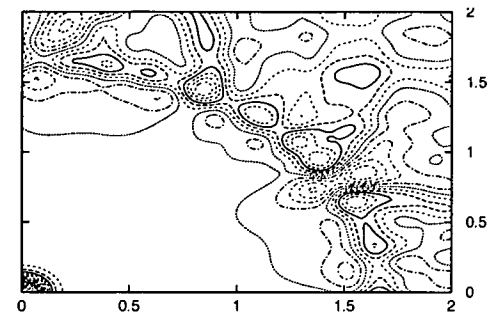

Figure 4.11: Contours of autocovariance function c_D for a process generated from a circular grid

periodic. Figure 4.13 shows the contours of a function which was generated using the circular grid as shown in Figure 4.7 with the same autocovariance function $c(\boldsymbol{x})$. It shows no obvious periodicity. The points $\boldsymbol{\omega}$ on the grids shown all satisfy $\|\boldsymbol{\omega}\| \leq 10$. Note that for $\|\boldsymbol{\omega}\| = 10$, $\tilde{c}(\boldsymbol{\omega}) = \pi \exp(-\|\boldsymbol{\omega}\|^2/4) \approx 4.363 \times 10^{-11}$ is small and terms involving $\|\boldsymbol{\omega}\| > 10$ are negligible.

The sample path generated from the circular grid displays no obvious periodicity, and therefore might make using the circular grid seem more appealing than the rectangular grid. However, neither grid gives a good approximation to the continuous autocovariance function $c(.)$ if the number of frequencies used is not large enough. Error estimates for the autocovariance function for the rectangular grid are relatively easy to obtain by extending the analysis for functions of one variable.

In all the cases reported in this thesis we choose the number of grid points large enough, and their spacing small enough, such that the error in the autocovariance of the discretely generated process is small. In what follows we use both the circular grid and the rectangular grid to generate functions. A similar spacing as for the rectangular grid is chosen for the circular grid, which seems likely also to give accurate approximations.

4.2.3 Higher Dimensions

So far we have discussed how to generate (stationary) Gaussian stochastic processes with a given autocovariance function, or at least a good approximation to it, in the 1-dimensional and the 2-dimensional case. Essentially the same method could in theory be used to generate Gaussian stochastic processes and sample paths of these in higher dimensions. In d dimensions, by analogy with (4.22), a stochastic process with autocovariance function $c_d(\boldsymbol{x})$ can be generated using

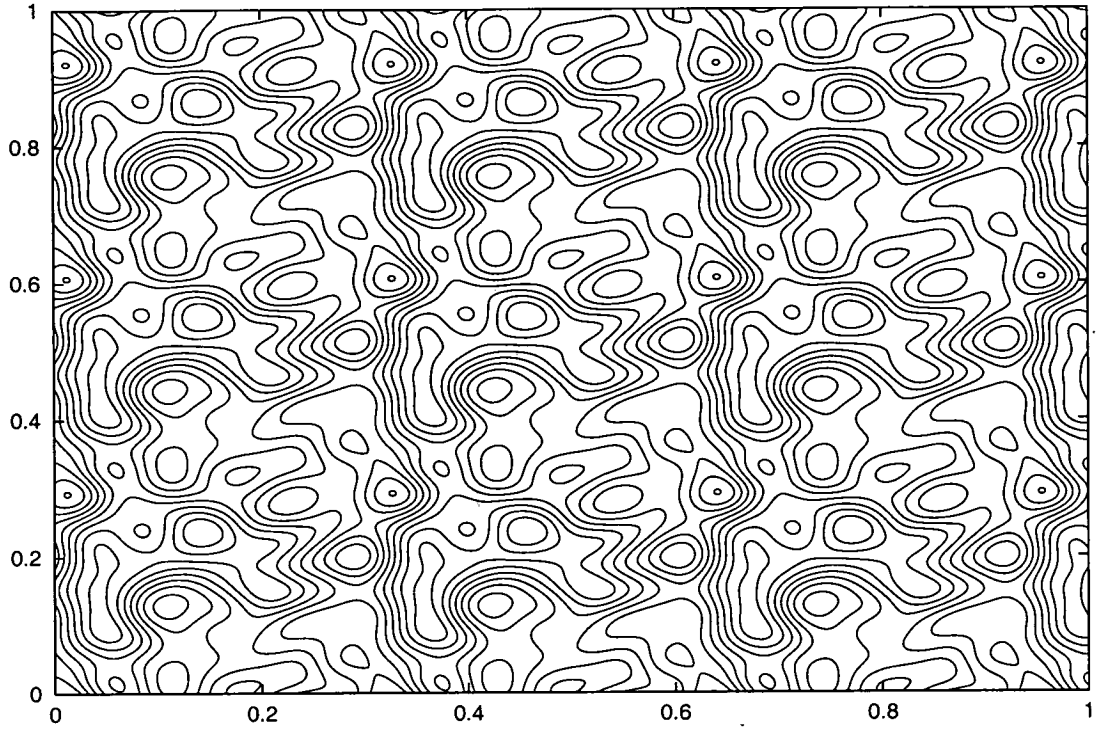

Figure 4.12: Function generated from a rectangular grid with 169 points and $\theta_1 = \theta_2 = 400$

$$Y(\boldsymbol{x}) = (2\pi)^{-d/2} \int_{\mathbb{R}^d} \sqrt{\tilde{c}_d(\boldsymbol{\omega})} L(\boldsymbol{\omega}) \cos((\boldsymbol{x}^t\boldsymbol{\omega}) + V(\boldsymbol{\omega})2\pi) d\boldsymbol{\omega}, \tag{4.25}$$

but there is the issue of the discretization to approximate the integral in (4.25). Using frequencies $\boldsymbol{\omega}$ from a regular grid is easily generalized to d dimensions, but since the number of frequencies increases to the power of d, the number of frequencies quickly becomes too large. Another option might be frequencies from a spherical grid, but generating a spherical grid in d dimensions is not straightforward. As an alternative to systematically dividing up the frequency space we could choose frequencies at random and use a Monte Carlo method. This seems to work well in the case of the bell-shaped exponential autocovariance function, and how this can be done, and how random processes can be generated from this will be explained in the following. In theory as the number of frequencies goes to infinity, the random processes we generate would have the bell-shaped exponential autocovariance function. Using a finite number of the random frequencies the process covariance function will only be approximately the bell-shaped exponential.

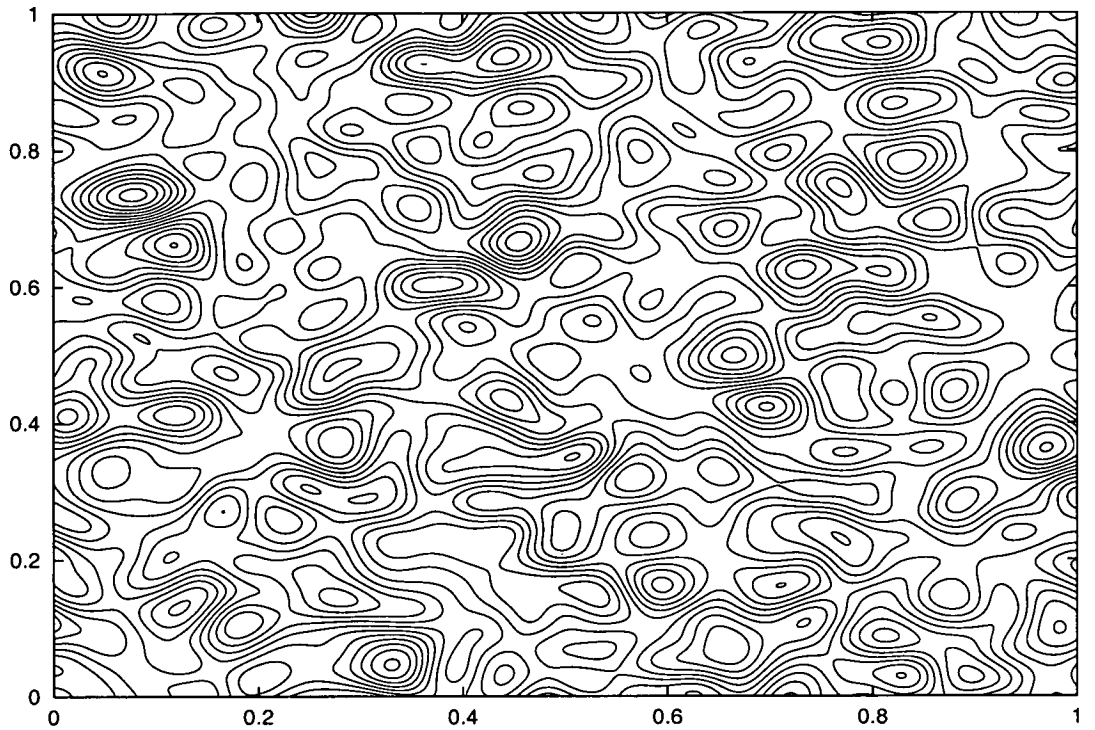

Figure 4.13: Function generated from a circular grid with 173 points and $\theta_1 = \theta_2 = 400$

Let the autocovariance function $c(.) : \mathbb{R}^d \to \mathbb{R}$ be the exponential autocovariance function

$$c(\boldsymbol{x}) = \exp\left(-||\boldsymbol{x}||^2\right).$$

The Fourier transform of $c(.)$ is

$$\begin{aligned} \tilde{c}(\boldsymbol{\omega}) &= \int_{\mathbb{R}^d} \exp\left(-i\boldsymbol{x}^t\boldsymbol{\omega}\right) c(\boldsymbol{x}) d\boldsymbol{x} \\ &= \pi^{d/2} \exp\left(-\frac{||\boldsymbol{\omega}||^2}{4}\right) \end{aligned}$$

and the inverse Fourier transform is

$$\begin{aligned} c(\boldsymbol{x}) &= (2\pi)^{-d} \int_{\mathbb{R}^d} \tilde{c}(\boldsymbol{\omega}) \exp\left(i\boldsymbol{x}^t\boldsymbol{\omega}\right) d\boldsymbol{\omega} \\ &= (2\pi)^{-d} \int_{\mathbb{R}^d} \pi^{d/2} \exp\left(-\frac{||\boldsymbol{\omega}||^2}{4}\right) \exp\left(i\boldsymbol{x}^t\boldsymbol{\omega}\right) d\boldsymbol{\omega} \\ &= (4\pi)^{-d/2} \int_{\mathbb{R}^d} \exp\left(-\frac{||\boldsymbol{\omega}||^2}{4}\right) \exp\left(i\boldsymbol{x}^t\boldsymbol{\omega}\right) d\boldsymbol{\omega}. \end{aligned}$$

Because of the symmetry properties of $\exp\left(-||\boldsymbol{\omega}||^2\right)$ this is

$$c(\boldsymbol{x}) = (4\pi)^{-d/2} \int_{\mathbb{R}^d} \exp\left(-\frac{||\boldsymbol{\omega}||^2}{4}\right) \cos\left(\boldsymbol{x}^t\boldsymbol{\omega}\right) d\boldsymbol{\omega}. \tag{4.26}$$

We can choose N frequencies $\boldsymbol{\omega}^{(1)}, \ldots, \boldsymbol{\omega}^{(N)}$ for our discretization at random, from the density function

$$f(\boldsymbol{\omega}) = (4\pi)^{-d/2} \exp\left(-\frac{||\boldsymbol{\omega}||^2}{4}\right)$$

so that each component of $\boldsymbol{\omega}^{(i)}$, $i = 1, \ldots, N$, is $N(0, 2)$ distributed. Now generate a stochastic process by

$$Y(\boldsymbol{x}) = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} (X_i \cos(\boldsymbol{x}^t \boldsymbol{\omega}^{(i)}) + Z_i \sin(\boldsymbol{x}^t \boldsymbol{\omega}^{(i)})) \tag{4.27}$$

or by

$$Y(\boldsymbol{x}) = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} L_i \cos(\boldsymbol{x}^t \boldsymbol{\omega}^{(i)} + V_i 2\pi).$$

using the Box-Muller transform.

To see the motivation for this, consider the autocovariance of $Y(\boldsymbol{x})$ and $Y(\boldsymbol{v})$ as in (4.27), this is

$$\begin{aligned} c_D(\boldsymbol{x} - \boldsymbol{v}) &= \operatorname{cov}(Y(\boldsymbol{x}), Y(\boldsymbol{v})) \\ &= E(Y(\boldsymbol{x})Y(\boldsymbol{v})) \\ &= \frac{1}{N} \sum_{i=1}^{N} \left(\cos(\boldsymbol{x}^t \boldsymbol{\omega}^{(i)}) \cos(\boldsymbol{v}^t \boldsymbol{\omega}^{(i)}) + \sin(\boldsymbol{x}^t \boldsymbol{\omega}^{(i)}) \sin(\boldsymbol{v}^t \boldsymbol{\omega}^{(i)})\right) \\ &= \frac{1}{N} \sum_{i=1}^{N} \cos((\boldsymbol{x} - \boldsymbol{v})^t \boldsymbol{\omega}^{(i)}). \end{aligned}$$

Now from the definition of the expectation and by (4.26)

$$\begin{aligned} E(\cos(\boldsymbol{x}^t \boldsymbol{\omega}^{(i)})) &= \int_{\mathbb{R}^d} (4\pi)^{-d/2} \exp\left(-\frac{||\boldsymbol{\omega}||^2}{4}\right) \cos(\boldsymbol{x}^t \boldsymbol{\omega}) d\boldsymbol{\omega} \\ &= \exp\left(-||\boldsymbol{x}||^2\right) \end{aligned}$$

and therefore the expected value of the autocovariance function of $Y(.)$ at any point $\boldsymbol{x} \in \mathbb{R}^d$ is

$$\begin{aligned} E(c_D(\boldsymbol{x})) &= \frac{1}{N} E\left(\sum_{i=1}^{N} \cos(\boldsymbol{x}^t \boldsymbol{\omega}^{(i)})\right) \\ &= \frac{N \exp\left(-||\boldsymbol{x}||^2\right)}{N} \\ &= \exp\left(-||\boldsymbol{x}||^2\right). \end{aligned}$$

Thus at any point $\boldsymbol{x} \in \mathbb{R}^d$ the expected value of the autocovariance function of the discrete process $Y(\boldsymbol{x})$ generated using (4.27) is the exponential autocovariance

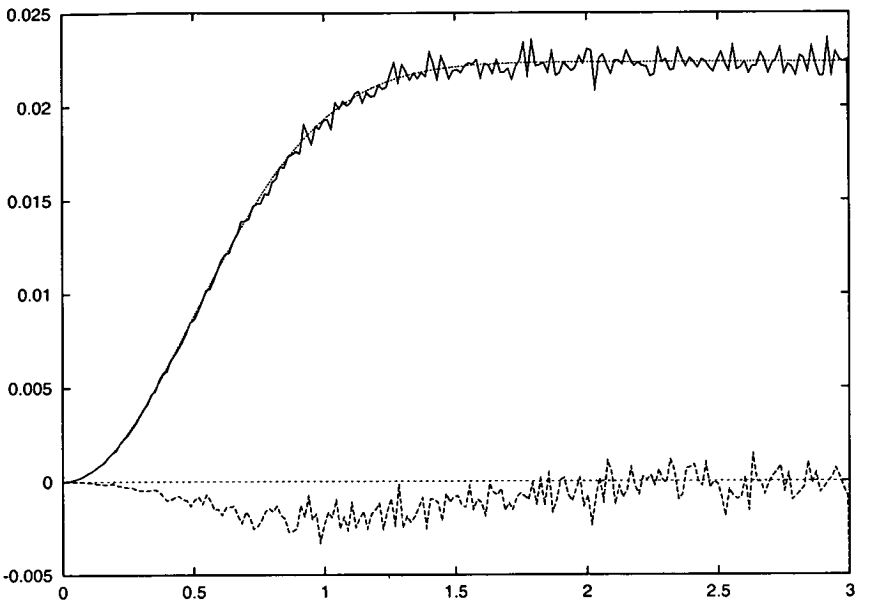

Figure 4.14: Autocovariance mean and variance

function $c(\boldsymbol{x}) = \exp(-||\boldsymbol{x}||^2)$, the autocovariance function we wanted $Y(\boldsymbol{x})$ to have. Now that we know the expected value of $c_D(\boldsymbol{x})$, can we estimate the error, i.e. the variance of $c_D(\boldsymbol{x})$? This will follow directly from the variances of the $\cos(\boldsymbol{x}^t\boldsymbol{\omega}^{(i)})$, $i = 1, \ldots, N$ by

$$\begin{aligned} \text{var}(c_D(\boldsymbol{x})) &= \text{var}\left(\frac{1}{N}\sum_{i=1}^{N}\cos((\boldsymbol{x}-\boldsymbol{v})^t\boldsymbol{\omega}^{(i)})\right) \\ &= \frac{1}{N^2}\sum_{i=1}^{N}\text{var}\left(\cos((\boldsymbol{x}-\boldsymbol{v})^t\boldsymbol{\omega}^{(i)})\right). \end{aligned} \tag{4.28}$$

The variance of $\cos(\boldsymbol{x}^t\boldsymbol{\omega}^{(i)})$ is given by

$$\text{var}(\cos(\boldsymbol{x}^t\boldsymbol{\omega}^{(i)})) = E(\cos(\boldsymbol{x}^t\boldsymbol{\omega}^{(i)}))^2 - (E(\cos(\boldsymbol{x}^t\boldsymbol{\omega}^{(i)})))^2$$

and

$$\begin{aligned} E(\cos(\boldsymbol{x}^t\boldsymbol{\omega}^{(i)}))^2 &= \int_{\mathbb{R}^d}(4\pi)^{-d/2}\exp\left(-\frac{||\boldsymbol{\omega}||^2}{4}\right)(\cos(\boldsymbol{x}^t\boldsymbol{\omega}))^2 d\boldsymbol{\omega} \\ &= (4\pi)^{-d/2}\int_{\mathbb{R}^d}\exp\left(-\frac{||\boldsymbol{\omega}||^2}{4}\right)\left(\frac{1}{2}\cos(2\boldsymbol{x}^t\boldsymbol{\omega})+\frac{1}{2}\right)d\boldsymbol{\omega} \\ &= \frac{1}{2}(4\pi)^{-d/2}\int_{\mathbb{R}^d}\exp\left(-\frac{||\boldsymbol{\omega}||^2}{4}\right)\left(\cos(2\boldsymbol{x}^t\boldsymbol{\omega})+1\right)d\boldsymbol{\omega} \\ &= \frac{1}{2}(4\pi)^{-d/2}\left[\int_{\mathbb{R}^d}\exp\left(-\frac{||\boldsymbol{\omega}||^2}{4}\right)\cos(2\boldsymbol{x}^t\boldsymbol{\omega})+\right. \\ &\qquad \left.\int_{\mathbb{R}^d}\exp\left(-\frac{||\boldsymbol{\omega}||^2}{4}\right)\right] \end{aligned}$$

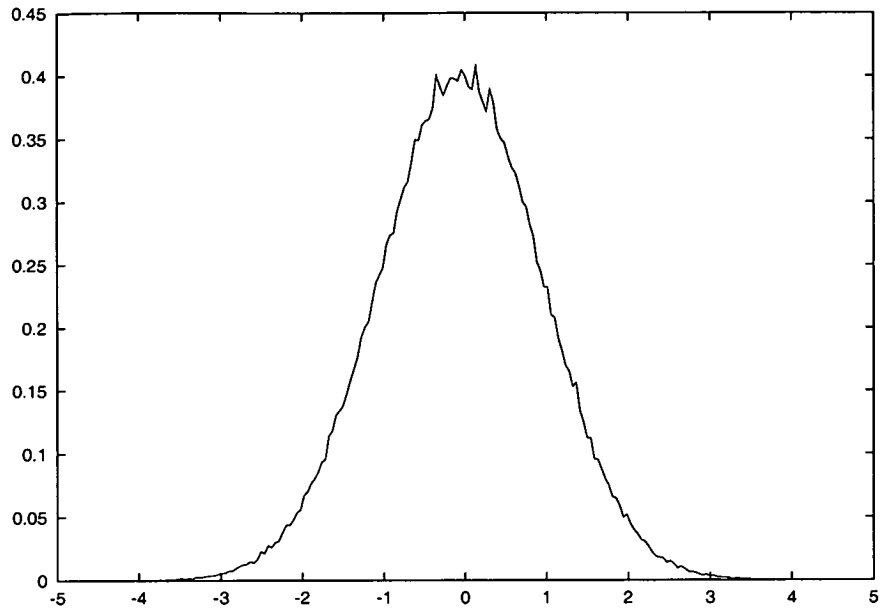

Figure 4.15: Error distribution

$$= \frac{1}{2} + \frac{1}{2}\exp\left(-4||\boldsymbol{x}||^2\right) d\boldsymbol{\omega}.$$

Therefore the variance of $\cos(\boldsymbol{x}^t\boldsymbol{\omega}^{(i)})$ is

$$\begin{aligned}\text{var}(\cos(\boldsymbol{x}^t\boldsymbol{\omega}^{(i)})) &= \frac{1}{2} + \frac{1}{2}\exp\left(-4||\boldsymbol{x}||^2\right) - \exp\left(-2||\boldsymbol{x}||^2\right) \\ &= \frac{1}{2}\left(1 - \exp\left(-2||\boldsymbol{x}||^2\right)\right)^2\end{aligned}$$

and it follows from (4.28) that the variance of $c_D(\boldsymbol{x})$ is

$$\text{var}(c_D(\boldsymbol{x})) = \frac{1}{2N}\left(1 - \exp\left(-2||\boldsymbol{x}||^2\right)\right)^2. \tag{4.29}$$

For small $||\boldsymbol{x}||$, $\exp\left(-2||\boldsymbol{x}||^2\right)$ is close to 1 and hence the variance is small. For large $||\boldsymbol{x}||$, $\exp\left(-2||\boldsymbol{x}||^2\right)$ is small, and the dominating term in (4.29) is $\frac{1}{2N}$. Therefore, for any $\boldsymbol{x}$ an upper bound on the variance is

$$\text{var}(c_D(\boldsymbol{x})) \leq \frac{1}{2N}.$$

Figure 4.14 shows an example with 1000 frequencies in 100 dimensions. Plotted are the theoretical expected error and standard deviation $\frac{1 - \exp(-2||\boldsymbol{x}||^2)}{\sqrt{2000}}$ and the expected error and standard deviation obtained using the 1000 frequencies. Figure 4.15 shows a corresponding error distribution, obtained by averaging over all $\boldsymbol{x}$ and scaling errors at $\boldsymbol{x}$ by $\frac{\sqrt{2N}}{1 - \exp(-2||\boldsymbol{x}||^2)}$.

4.2.4 Mean and Variance

If $Y(\boldsymbol{x})$ has been generated from the autocorrelation function $\rho(.)$ by the above methods, and has mean $\mu = 0$ and variance $\sigma^2 = 1$, and if we set

$$Z(\boldsymbol{x}) = \sigma Y(\boldsymbol{x}) + \mu,$$

then the expectation of $Z(\boldsymbol{x})$ is $E(Z(\boldsymbol{x})) = \mu$ and the covariance of $Z(\boldsymbol{x})$ and $Z(\boldsymbol{v})$ by (3.1.2) is

$$\begin{aligned} \operatorname{cov}(Z(\boldsymbol{x}), Z(\boldsymbol{v})) &= E[(Z(\boldsymbol{x}) - \mu)(Z(\boldsymbol{v}) - \mu)] \\ &= E[\sigma Z(\boldsymbol{x}) \sigma Z(\boldsymbol{v})] \\ &= \sigma^2 E[Z(\boldsymbol{x}) Z(\boldsymbol{v})] \\ &= \sigma^2 \rho(\boldsymbol{x} - \boldsymbol{v}). \end{aligned}$$

4.3 Conditional Functions

Here the 1-dimensional case is considered only, but it extends to higher dimensions in much the same way as function generating described previously. Suppose that in the process of optimizing the objective function we have sampled at n points $x^{(1)}, \ldots, x^{(n)}$ and the vector of function values is $\boldsymbol{y} = (y_1, \ldots, y_n)^t$. We might be interested in possible functions with certain parameter values for θ, μ, and σ^2, that interpolate the objective function in the known data points $(x^{(1)}, y_1), \ldots, (x^{(n)}, y_n)$. These functions can be used for example to tell where a sample path with the given parameter values, and which interpolates the data, is most likely to attain its minimum. Here we will show how to generate stochastic processes and sample paths, conditional on the n points. Suppose the autocovariance c is approximated by

$$c_D(x) = \sum_k \tilde{c}(\omega^{(k)}) \cos(x\omega^{(k)}).$$

As before the unconditional $Y(x)$ with autocovariance $c_D(x)$ can be generated by

$$Y(x) = \frac{1}{\sqrt{\pi}} \sum_{k=0}^{\infty} \sqrt{a_k} \sqrt{\tilde{c}(\omega^{(k)})} L_k \cos(x\omega^{(k)} + V_k 2\pi),$$

and we can use $Y(x)$ to generate $Z(x)$ conditional on $Z(x_j) = y_j$ for $j = 1, \ldots, n$. as follows: let R be the $n \times n$ autocorrelation or autocovariance matrix with entries $R_{ij} = c_D(x^{(i)} - x^{(j)})$. Let $\boldsymbol{a}$ be the n-vector with entries $a_k = y_k - Y(x^{(k)})$, and let $\boldsymbol{b} = R^{-1}\boldsymbol{a}$, so $a_k = \sum_j R_{kj} b_j$. Then letting

$$Z(x) = Y(x) + \sum_{i,j} b_j \tilde{c}(\omega^{(i)}) \cos((x - x^{(j)})\omega^{(i)}),$$

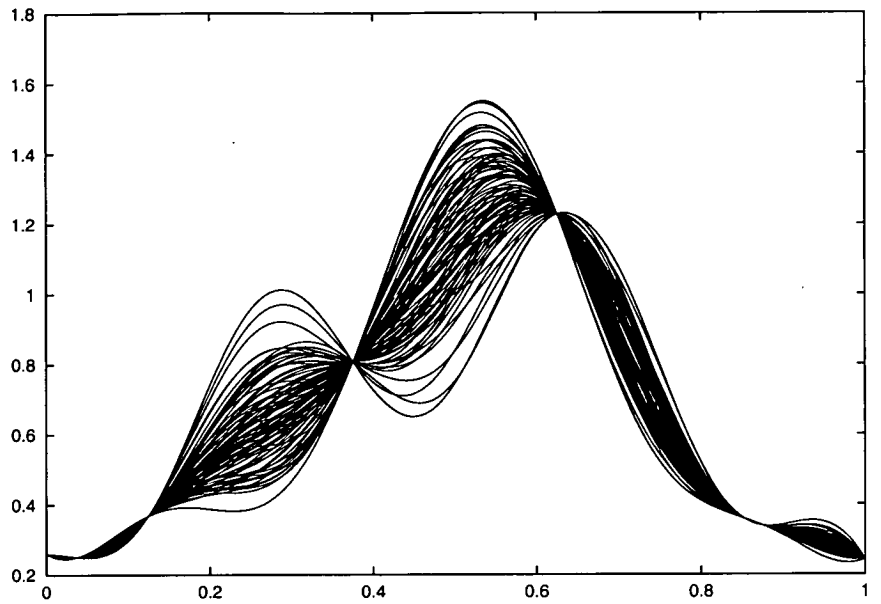

Figure 4.16: Conditional functions using maximum likelihood estimates of θ, μ, and σ

$Z(x)$ has the required interpolating property, since

$$\begin{aligned}
Z(x^{(k)}) &= Y(x^{(k)}) + \sum_{i,j} b_j \tilde{c}(\omega^{(i)}) \cos((x^{(k)} - x^{(j)})\omega^{(i)}) \\
&= Y(x^{(k)}) + \sum_{j} b_j \sum_{i} \tilde{c}(\omega^{(i)}) \cos((x^{(k)} - x^{(j)})\omega^{(i)}) \\
&= Y(x^{(k)}) + \sum_{j} b_j c_D(x^{(k)} - x^{(j)}) \\
&= Y(x^{(k)}) + \sum_{j} b_j R_{kj} \\
&= Y(x^{(k)}) + a_k \\
&= Y(x^{(k)}) + y_k - Y(x^{(k)}) \\
&= y_k.
\end{aligned}$$

By properties of the Gaussian distribution, the conditional functions generated by this method also have Gaussian distribution.

Figure 4.16 shows an example of 100 conditional functions. The data points are the final set of sample points from one of our test problems on which the optimization algorithm misses the global optimum value by more than 0.1 and finds a local optimum at $x = 1$, where the actual global optimum is at $x = 0.2417$. The parameters θ, μ, σ are the maximum likelihood estimates given the available data points and the 100 plotted functions all have these values for θ, μ, σ. For comparison, Figure 4.17 shows the BLUP $\hat{y}(x)$ plus-minus three

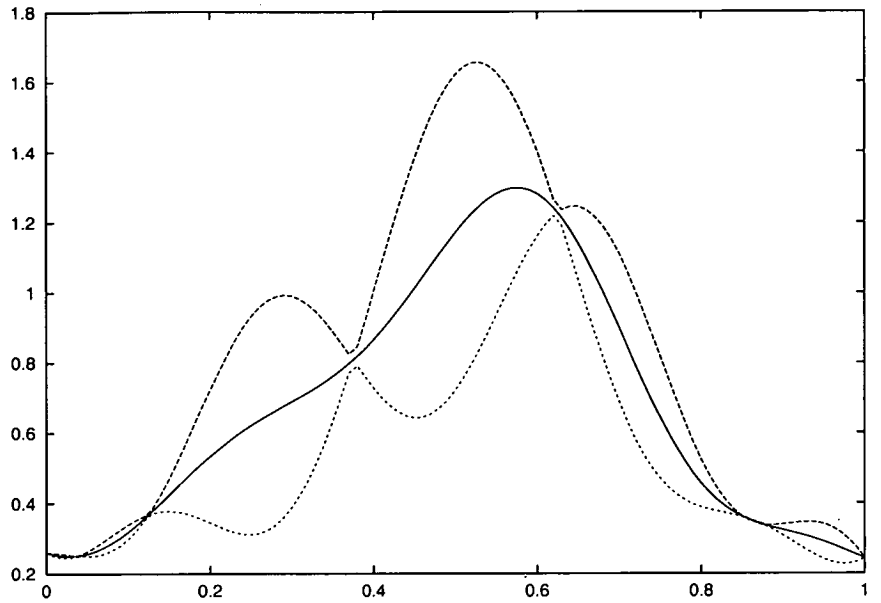

Figure 4.17: Confidence interval, $\hat{y}(x) \pm 3s(x)$

standard deviations $s(x)$, i.e. $\hat{y}(x) \pm 3s(x)$. The $\hat{y}(x)$ is the posterior mean and the plot essentially shows a confidence interval for the function values at the points $x \in [0, 1]$ for the given parameters θ, μ, σ. We would expect most parts of most of the possible functions with these parameter values to lie within the outer curves plotted in Figure 4.17, and this is the case here.

4.4 Priors and Posteriors

There are several possible ways of addressing the problem of modelling the objective function. The model can involve parameters which can be fixed, or estimated by some means. So one way of modeling the function is to assume that the stochastic process which the function is assumed to be a realization of is known entirely. In this case there are no parameters to be estimated. Another possibility is to estimate certain parameters from available data, for example by maximum likelihood estimation. A third possibility is to allow for possible uncertainty in the parameters. This then leads to the Bayesian approach, where prior distributions are assumed or known for the parameters, which are then used to obtain posterior distributions conditional on available function data. The issue of prior and posterior distributions will be addressed in this section.

To start with we describe a family of probability density functions which can be used both as prior and posterior probability density functions for θ, σ and μ. The form of these functions has several advantages:

- It is easy to generate samples from these distributions.
- If the prior probability is of this form then so too is the posterior probability. This simplifies a Bayesian approach to updating the parameters of the stochastic process.
- It is possible to find marginal distributions by integrating analytically.

The distributions are defined in terms of the random variables τ, $\bar{\sigma}$, μ, where

$$\tau = \frac{1}{\sqrt{\theta}}, \quad \bar{\sigma} = \frac{1}{\sigma}, \tag{4.30}$$

and the density function $p(\tau, \bar{\sigma}, \mu)$ can be written in the form

$$p(\tau, \bar{\sigma}, \mu)d\tau d\bar{\sigma}d\mu = (p_\tau(\tau)d\tau)(p_{\bar{\sigma}}(\bar{\sigma}|\tau)d\bar{\sigma})(p_\mu(\mu|\tau, \bar{\sigma})d\mu).$$

To obtain a sample from a distribution of this form, we first sample τ from p_τ, then $\bar{\sigma}$ from $p_{\bar{\sigma}}$, and then μ from p_μ, and finally find θ and σ from (4.30). For both prior and posterior distributions p_μ is a normal and $p_{\bar{\sigma}}$ is a G distribution, as in Definition 3.1.9.

4.4.1 Prior Distributions

We assume that μ has a normal prior distribution, $\mu \sim N\left(0, \frac{1}{a^2\bar{\sigma}^2}\right)$, i.e. it has probability density function

$$p_\mu(\mu|\tau, \bar{\sigma}) = (2\pi)^{-\frac{1}{2}} a\bar{\sigma} \exp\left(-\frac{a^2\bar{\sigma}^2\mu^2}{2}\right),$$

where a is a function of τ only. Note that the mean of μ is zero, and the standard deviation of μ is proportional to σ.

For $\bar{\sigma}$ we assume a $G(\gamma, b^2)$ distribution as a prior distribution, i.e. $\bar{\sigma}$ has probability density function

$$p_{\bar{\sigma}}(\bar{\sigma}|\tau) = K_\gamma b^\gamma \bar{\sigma}^{\gamma-1} \exp\left(-\frac{b^2\bar{\sigma}^2}{2}\right), \tag{4.31}$$

where b is function of τ only.

The p_τ posterior distribution has no simple closed form so deviates of that distribution have to be calculated numerically. Because of this there is no advantage in taking any special form for the prior p_τ.

One family of problems has been generated using a $G(4, 2 \times 12^2)$ distribution for τ, a $G(4, 2)$ distribution for $\bar{\sigma}$, i.e. $b^2 = 2$ in (4.31), and a $N\left(0, \frac{1}{\bar{\sigma}^2}\right)$ for μ, i.e. $a = 1$ in (4.31), a constant independent of τ.

4.4.2 Posterior Distributions

Once the values $\boldsymbol{y}$ of the objective function are known at a set of sample points $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}\}$ we can calculate the posterior distribution of the parameters τ, $\bar{\sigma}$ and μ. This has the form

$$p(\tau, \bar{\sigma}, \mu|\boldsymbol{y}) = K_1 p_\tau(\tau) p_{\bar{\sigma}}(\bar{\sigma}|\tau) p_\mu(\mu|\tau, \bar{\sigma}) L(\boldsymbol{y}|\tau, \mu, \bar{\sigma}) \tag{4.32}$$

where L is the likelihood function, i.e. the probability of the data given the parameters. The K_1 is a constant to normalise the cumulative probability to 1, and in what follows K_i will denote a similar constant.

From (3.18) we know that L has the form

$$L(\boldsymbol{y}) = (2\pi)^{-n/2} |R|^{-1/2} \bar{\sigma}^n \exp\left(-\frac{\bar{\sigma}^2 (\boldsymbol{y} - \mathbf{1}\mu)^t R^{-1} (\boldsymbol{y} - \mathbf{1}\mu)}{2}\right), \tag{4.33}$$

and R is the correlation or covariance matrix, which depends on τ and $\boldsymbol{x}$ only.

We wish to express this posterior probability in the following form

$$p(\tau, \bar{\sigma}, \mu|\boldsymbol{y}) = p_\tau(\tau|\boldsymbol{y}) p_{\bar{\sigma}}(\bar{\sigma}|\tau, \boldsymbol{y}) p_\mu(\mu|\tau, \bar{\sigma}, \boldsymbol{y}), \tag{4.34}$$

which will allow us to obtain samples as described earlier by successively sampling τ, then $\bar{\sigma}$ then μ.

Substituting for the prior $p_\mu(\mu|\tau, \bar{\sigma})$ and the likelihood function $L(\boldsymbol{y}|\tau, \mu, \bar{\sigma})$ in (4.32) gives

$$\begin{aligned}
p(\tau, \bar{\sigma}, \mu|\boldsymbol{y}) &= K_2 p_\tau(\tau) p_{\bar{\sigma}}(\bar{\sigma}|\tau) a\bar{\sigma} \exp\left(-\frac{a^2\bar{\sigma}^2\mu^2}{2}\right) \\
&\qquad |R|^{-1/2} \bar{\sigma}^n \exp\left(-\frac{\bar{\sigma}^2 (\boldsymbol{y} - \mathbf{1}\mu)^t R^{-1} (\boldsymbol{y} - \mathbf{1}\mu)}{2}\right) \\
&= K_2 p_\tau(\tau) a |R|^{-1/2} p_{\bar{\sigma}}(\bar{\sigma}|\tau) \bar{\sigma}^{n+1} \\
&\qquad \exp\left(-\frac{\bar{\sigma}^2 (a^2\mu^2 + (\boldsymbol{y} - \mathbf{1}\mu)^t R^{-1} (\boldsymbol{y} - \mathbf{1}\mu))}{2}\right) \\
&= K_2 p_\tau(\tau) a |R|^{-1/2} p_{\bar{\sigma}}(\bar{\sigma}|\tau) \bar{\sigma}^{n+1} \\
&\qquad \exp\left(-\frac{\bar{\sigma}^2 ((a^2 + \mathbf{1}^t R^{-1} \mathbf{1})\mu^2 - 2\mathbf{1}^t R^{-1} \boldsymbol{y}\mu + \boldsymbol{y}^t R^{-1} \boldsymbol{y})}{2}\right) \\
&= K_2 p_\tau(\tau) a |R|^{-1/2} p_{\bar{\sigma}}(\bar{\sigma}|\tau) \bar{\sigma}^{n+1} \\
&\qquad \exp\left(-\frac{\bar{\sigma}^2 \lambda^2}{2}\right) \exp\left(-\frac{\bar{\sigma}^2 \breve{\sigma}^2 (\mu - \breve{\mu})^2}{2}\right),
\end{aligned}$$

where

$$\begin{aligned}
\breve{\sigma}^2 &= a^2 + \mathbf{1}^t R^{-1} \mathbf{1} \\
\breve{\mu} &= \frac{\mathbf{1}^t R^{-1} \boldsymbol{y}}{\breve{\sigma}^2} \\
\lambda^2 &= \boldsymbol{y}^t R^{-1} \boldsymbol{y} - \breve{\sigma}^2 \breve{\mu}^2.
\end{aligned}$$

Note that $\breve{\sigma}$, $\breve{\mu}$ and λ depend only on τ. It follows that for fixed τ and $\bar{\sigma}$, μ is normally distributed with mean $\breve{\mu}$ and variance $1/(\breve{\sigma}\bar{\sigma})^2$.

Now integrating this from $-\infty$ to ∞ with respect to μ, we get the marginal posterior distribution with respect to τ and $\bar{\sigma}$

$$p(\tau, \bar{\sigma}|\boldsymbol{y}) = K_3 p_\tau(\tau) a |R|^{-1/2} \breve{\sigma}^{-1} p_{\bar{\sigma}}(\bar{\sigma}|\tau) \bar{\sigma}^n \exp\left(-\frac{\bar{\sigma}^2\lambda^2}{2}\right).$$

If $p_{\bar{\sigma}}(\bar{\sigma}|\tau)$ is a $G(\gamma, b^2)$ distribution (see (4.31)) then

$$\begin{aligned} p(\tau, \bar{\sigma}|\boldsymbol{y}) &= K_4 p_\tau(\tau) a |R|^{-1/2} \breve{\sigma}^{-1} b^\gamma \bar{\sigma}^{\gamma-1} \exp\left(-\frac{b^2\bar{\sigma}^2}{2}\right) \bar{\sigma}^n \exp\left(-\frac{\bar{\sigma}^2\lambda^2}{2}\right) \\ &= K_4 p_\tau(\tau) a |R|^{-1/2} \breve{\sigma}^{-1} b^\gamma \bar{\sigma}^{\gamma+n-1} \exp\left(-\frac{\bar{\sigma}^2 (b^2+\lambda^2)}{2}\right), \end{aligned}$$

so for fixed τ, $\bar{\sigma}$ has a $G(\gamma + n, b^2 + \lambda^2)$ distribution.

Now integrating this from 0 to ∞ with respect to $\bar{\sigma}$ we get the marginal posterior distribution with respect to τ:

$$p(\tau|\boldsymbol{y}) = K_5 p_\tau(\tau) a |R|^{-1/2} \breve{\sigma}^{-1} b^\gamma (b^2+\lambda^2)^{-(\gamma+n)/2}.$$

Here all the terms are functions of τ only. Samples of τ from this distribution can be found by calculation an approximation to its cumulative distribution function.

4.5 Test Sets

The functions used as test functions in the following chapters are functions generated by the methods described earlier in this chapter. These sets of problems consist of functions generated with the autocorrelation function $\rho(x) = \exp(-\theta x^2)$. One set of 500 functions is generated with $\theta = 25$, $\mu = 0$ and $\sigma = 1$, and this set is referred to as T025. Another set of 500 functions is generated with $\theta = 225$, $\mu = 0$ and $\sigma = 1$, which is referred to as T225. To reflect real life situations more accurately where the parameters θ, μ and σ are unknown, another set of 500 test functions is generated with the parameters θ, μ and σ all chosen at random from prior distributions as described in the previous Section. This set of functions is denoted by Txxx. Figure 4.18 shows 100 such functions, with the parameters θ, μ, σ sampled from the above distributions, and Figure 4.19 shows the distribution of $\tau = \frac{1}{\sqrt{\theta}}$ for the sample paths plotted in Figure 4.18.

These three sets of functions described here, each consisting of 500 functions, are the main test functions used.

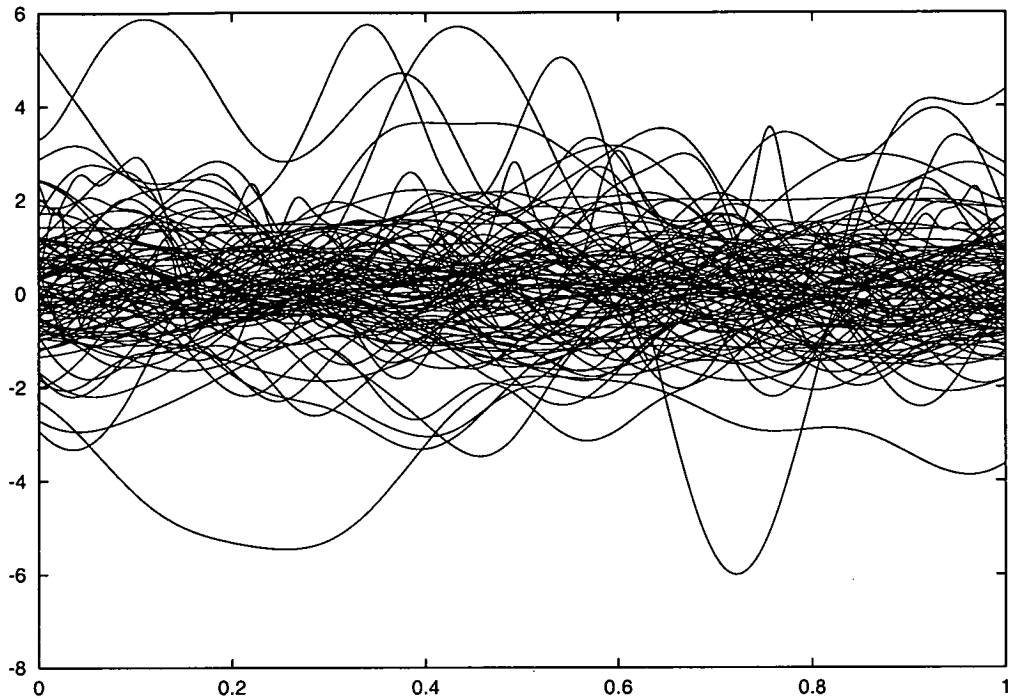

Figure 4.18: Sample paths

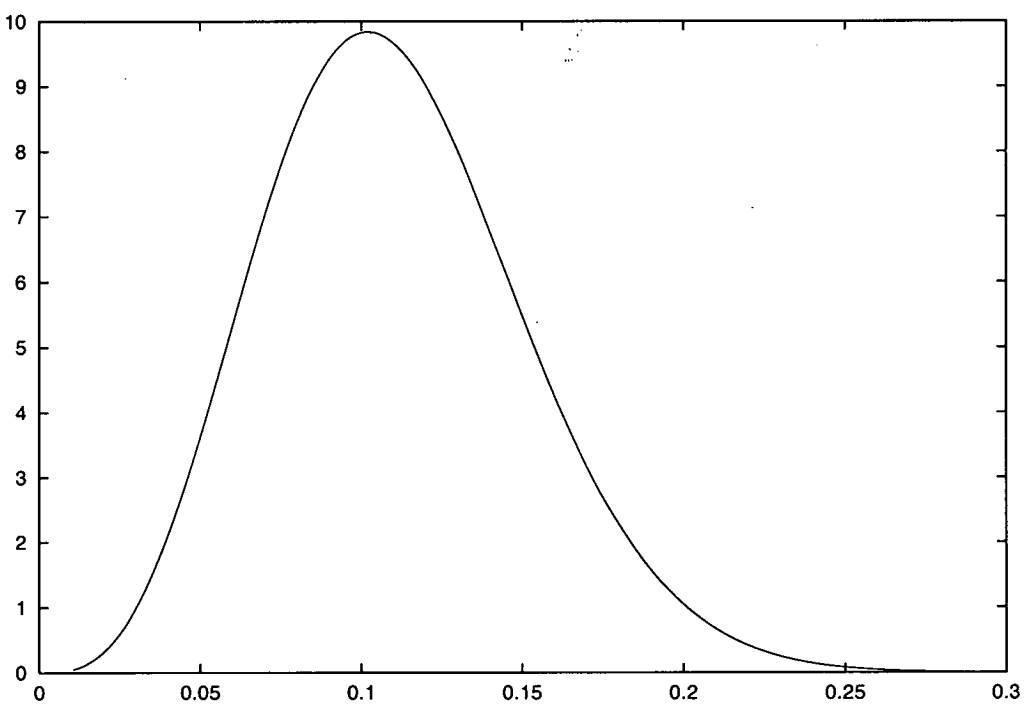

Figure 4.19: Prior distribution for $\tau = 1/\sqrt{\theta}$

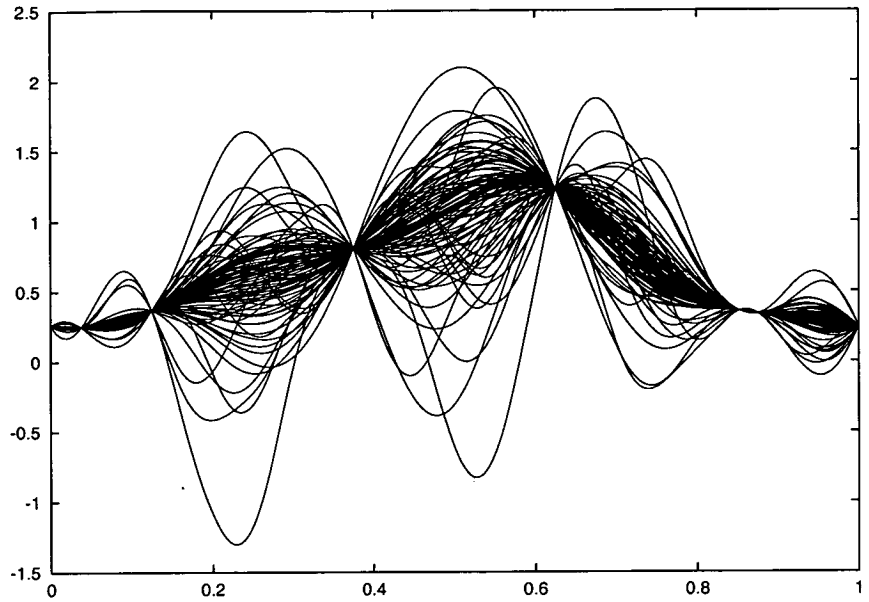

Figure 4.20: Conditional functions with parameters sampled from their posterior distributions

4.6 Conclusions

The generated processes are (stationary) Gaussian stochastic processes with specified autocorrelation or autocovariance, mean, and variance. Particular outcomes or sample paths of these Gaussian stochastic processes are obtained here by choosing a set of frequencies and outcomes for the random variables in the generating process. These sample paths can be expected to have the same properties as the stochastic process, and this makes them useful as test functions since they are of the form assumed in the theory of the optimization algorithm. This helps in monitoring the progress and performance of the algorithm in the optimization process. The generation of test problems like this can be automated, and generating large numbers of test functions is relatively easy. This allows us to run our algorithm on a large number of test problems and to make a reliable comparison between different optimization algorithms, and get reliable statistics on the proportion of runs that fail and the reasons for this.

Figure 4.20 suggests that taking into account the uncertainty in the estimation of the parameters θ, μ, and σ could save the algorithm from terminating prematurely which occurs when using a maximum expected improvement based on a single estimate of the parameters. An approach which makes use of this observation is introduced in Chapter 5.

Chapter 5

Implementation

In this chapter certain features of our implementation are introduced. Our algorithm is based on the Kriging approach and the expected improvement criterion as described in Chapter 3. This method brings with it certain problems and drawbacks which we will discuss here, along with possible ways of addressing them. As test functions we use functions generated as described in Chapter 4.

A potential problem in this method of optimization is near-singularity of the correlation matrix. This can occur if $\boldsymbol{\theta}$ is small or if two or more sample points are too close together. We briefly explain different methods by which this problem can be addressed.

A possible way of maximizing the expected improvement function by branch and bound was described in Chapter 3. This requires finding upper bounds on the expected improvement in given boxes. The bounding can be done by separately minimizing the BLUP $\hat{y}(\boldsymbol{x})$ and maximizing the MSE $s^2(\boldsymbol{x})$, or equivalently minimizing $-s^2(\boldsymbol{x})$, subject to non-linear constraints. The BLUP is a linear function of $\boldsymbol{r}$, and the nonlinear constraints, by which $\boldsymbol{r}$ is related to $\boldsymbol{x}$, can be linearized, but the question remains whether $-s^2(\boldsymbol{x})$ is convex in $\boldsymbol{r}$. If $-s^2(\boldsymbol{x})$ is convex then, subject to the linearized constraints, the resulting problem is a convex QP. Jones *et al.* [34] leave open the possibility that the function $-s^2(\boldsymbol{x})$ is nonconvex, and use the αBB branch and bound method, which convexifies the objective function if necessary, to solve this potentially non-convex optimization problem. Details of αBB can be found for example in Andraloukis *et al.* [3]. It is shown in this chapter that if the $n \times n$ correlation matrix R is positive definite, and if only constant regression terms are used, then the Hessian matrix of $-s^2(\boldsymbol{x})$ is positive semi-definite, and therefore the minimization of $-s^2(\boldsymbol{x})$ is easier than previously thought.

Bounding the BLUP and MSE separately to find an upper bound on the expected improvement typically yields tight bounds on the expected improvement only when the box B is small. This is because minima of the BLUP and maxima

of the MSE do not usually occur at the same point $\boldsymbol{x} \in D$. A possible better bounding procedure will be discussed in this chapter.

As described in Chapter 3, Jones *et al.* [34] not only use a convexification for the MSE function but also approximate the nonlinear equations by linear over- and under-estimators. These are not tight and in this chapter it is shown that there is considerable scope for improvement in the linear approximations of these equations.

In the maximization of the expected improvement function, typically the branch and bound requires a lot of branching before useful upper or lower bounds on the expected improvement are found. It can be expected that the number of boxes examined will be reduced significantly by finding tighter upper bounds on the expected improvement or tighter linearizations of the constraints. However, it might not be necessary to use a global optimization method like branch and bound. As an alternative to branch and bound we also use a simple grid search plus local search.

A problem with the type of response surface optimization method we are investigating here, is what is called *overfitting* in the estimation of the model parameters. The sample can be deceptive and $\boldsymbol{\theta}$ estimated to be considerably smaller than would be appropriate for the objective function. As a consequence the algorithm might stop prematurely without even having got near a local minimum. Similarly, estimates of μ and σ might not match the objective function well and can lead to failure of the algorithm.

5.1 Near-Singularity and Regularization

Here we briefly address the problem of near-singularity in the correlation matrix R and possible ways of overcoming it. Some ways of dealing with near-singularity have been suggested in the global optimization literature.

5.1.1 Near-Singularity

When two or more sample points are very close together, the matrix R has, respectively, two rows and columns which are near-dependent. When the entries of $\boldsymbol{\theta}$ are very small, all entries of R are close to unity. In both of these situations the matrix R is near-singular, in which case operations involving R^{-1} will be ill-conditioned and a source of numerical error when using inexact arithmetic. The estimation of the parameters $\boldsymbol{\theta}$ by maximum likelihood requires operations with R^{-1}, and if this is ill-conditioned numerical errors are introduced into the calculations and the estimates of the parameters themselves become less reliable.

In the parameter estimation we therefore avoid values of $\boldsymbol{\theta}$ which lead to a near-singular correlation matrix R, by putting a large penalty on parameters which lead to a near-singular matrix R. So near-singularity will occur only if points are sampled very close together, for example when homing in on a local optimum. In the following we will introduce some remedies to the problem of R becoming near-singular.

5.1.2 Remedies

In trying to find remedies for the problem of near-singularity of the correlation matrix R we face a trade-off. While R is near-singular, either because two sample points are too close together or elements of $\boldsymbol{\theta}$ are very small, we will lose out because of the ill-conditioning of the systems we have to solve. We can omit some of the information we have about the objective function to improve the conditioning. For example a sample point could be omitted, but since evaluations of the objective function are expensive we do not want to drop too much information. Essentially we want to improve the conditioning of the correlation matrix R by dropping or altering the rows or columns that lead to linear dependencies, but at the same time the aim is to lose as little valuable information about the objective function as possible. In particular we want to avoid resampling at or near a point which has been omitted.

5.1.2.1 Omitting Eigenvalues

In order to perform operations with R^{-1} it is best to factorize the matrix R in some way. One possible factorization, unfortunately an expensive one, is the decomposition using eigenvectors and eigenvalues of the matrix. Let $\lambda_1, \ldots, \lambda_n$ denote the eigenvalues of R and $\boldsymbol{v}^{(1)}, \ldots, \boldsymbol{v}^{(n)}$ denote the corresponding eigenvectors. The eigenvectors $\boldsymbol{v}^{(1)}, \ldots, \boldsymbol{v}^{(n)}$ are computed such that they are orthonormal. It is shown in (5.1) that the matrix R is positive definite, therefore $\lambda_i > 0$ for $i = 1, \ldots, n$, but if R is almost singular, some of the eigenvalues will be very small. We assume here, without loss of generality, that the eigenvalues are ordered by magnitude so that $\lambda_1 \geq \ldots \geq \lambda_n$. Let Σ be the diagonal matrix with entries $\lambda_1 \geq \ldots \geq \lambda_n$ and let V be the matrix with the eigenvectors $\boldsymbol{v}^{(1)}, \ldots, \boldsymbol{v}^{(n)}$ as its columns. Since $\boldsymbol{v}^{(1)}, \ldots, \boldsymbol{v}^{(n)}$ are orthonormal the matrix V is orthogonal. Then R can be written as

$$\begin{aligned} R &= V \Sigma V^t \\ &= \sum_{k=1}^{n} \lambda_k \boldsymbol{v}^{(k)} \boldsymbol{v}^{(k)t}. \end{aligned}$$

Since V is orthogonal,

$$R^{-1} = V\Sigma^{-1}V^t$$

where the diagonal matrix Σ^{-1} has entries $1/\lambda_1, \ldots, 1/\lambda_n$. If any of the eigenvalues are very small, say smaller than a tolerance δ, we set the corresponding entry in Σ^{-1} to zero. So if $\lambda_i, \ldots, \lambda_n \leq \delta$ then we set the pseudo-inverse Σ^+ of Σ to have diagonal entries $1/\lambda_1, \ldots, 1/\lambda_{i-1}, 0, \ldots, 0$ and let

$$\begin{aligned} R^+ &= V\Sigma^+V^t \\ &= \sum_{k=1}^{i-1} \frac{1}{\lambda_k} \boldsymbol{v}^{(k)}\boldsymbol{v}^{(k)t}. \end{aligned}$$

Note that (1) for $q < r = \text{rank}(R)$ the matrix

$$R_q = \sum_{k=1}^{q} \lambda_k \boldsymbol{v}^{(k)}\boldsymbol{v}^{(k)t}$$

is the matrix of rank q which is closest to R with respect to the 2-norm, i.e.

$$\min_{\text{rank}(A)=q} ||R - A||_2 = ||R - R_q||_2,$$

and note that (2) the least squares problem of minimizing $||R\boldsymbol{z} - \boldsymbol{b}||_2$ with (near) rank-deficient matrix R has the solution $\boldsymbol{z}_{LS} = R^+\boldsymbol{b}$ and R^+ is the unique solution of

$$\min_{A \in \mathbb{R}^{n \times n}} ||RA - I_n||_F$$

where

$$||R||_F = \sqrt{\sum_{j,k=1}^{n} |R_{ij}|^2}.$$

For details of this see for example Golub and Van Loan's book [26]. The properties (1) and (2) above mean that R^+ is the "best" pseudo-inverse of the matrix R, and it can be used as such when R becomes near-singular.

5.1.2.2 Perturbation

One way to solve systems involving the correlation matrix R is to find its Cholesky factorization $R = LL^t$, where L is a lower triangular matrix. The systems can then be solved by forward and backward substitution. If the matrix R is near-singular, one or more diagonal entries of the Cholesky factors can be very small leading to numerical growth in the factors. This may be prevented by perturbing small diagonal entries of the Cholesky factor L as they are encountered. The resulting factorization then corresponds to a perturbation of the matrix R.

5.1.2.3 Regularization

Lophaven, Nielsen, and Sondergaard in [44, 45] consider the *regularized* matrix $\hat{R} = R + \zeta I$ with $\zeta > 0$, for which $\hat{R}\boldsymbol{v}^{(i)} = (\lambda_i + \zeta)\boldsymbol{v}^{(i)}$, $i = 1, \ldots, n$. Therefore $\hat{R}$ has the same eigenvectors as R and the eigenvalues of $\hat{R}$ are the eigenvalues of R increased by ζ. In particular in [44] Lophaven, Nielsen, and Sondergaard suggest using $\zeta = (10 + n)\epsilon_M$ where ϵ_M is the machine precision.

5.1.2.4 Leaving out points

If at some stage in the algorithm two or more sample points are very close together this can lead to ill-conditioning of the matrix R. It seems justified to think that if we drop the point with the worst function value and keep the one or more points in that region with better values, we can improve the conditioning of the matrix R without losing too much valuable information about the objective function. Only sample points very close to other sample points with better objective function value would be left out, and it can be hoped for/expected that the BLUP will not be compromised too much and we will not sample the point we have just discarded again. However, this would require further tests in practice. One way of implementing this is whenever the Euclidean distance of two points is too small, to just drop the one with the higher function value.

5.2 Convexity

Let us first consider the definiteness of the correlation matrix R. According to Searle in [67] it is justified to assume non-negative definiteness for a variance-covariance and therefore a correlation matrix. Lophaven *et al.* in [44] also address the issue of the definiteness of R, this is the approach which we will follow here. Let us remind ourselves that previously we were looking at the objective function values $Y = (Y(\boldsymbol{x}^{(1)}), \ldots, Y(\boldsymbol{x}^{(n)}))^t$ for sample points $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$, and the stochastic process model

$$Y(\boldsymbol{x}) = F\boldsymbol{\beta} + \boldsymbol{\epsilon}(\boldsymbol{x})$$

where $\boldsymbol{\epsilon}(\boldsymbol{x})$ is assumed to have covariance matrix $\sigma^2 R$. Let $\boldsymbol{\zeta} \in \mathbb{R}^n$ and consider

$$\boldsymbol{\zeta}^T Y(\boldsymbol{x}) = \boldsymbol{\zeta}^T F\boldsymbol{\beta} + \boldsymbol{\zeta}^T \boldsymbol{\epsilon}(\boldsymbol{x}).$$

Then

$$\begin{aligned} E(\boldsymbol{\zeta}^T \boldsymbol{\epsilon}(\boldsymbol{x}) \boldsymbol{\zeta}^T \boldsymbol{\epsilon}(\boldsymbol{x})) &= \boldsymbol{\zeta}^T E(\boldsymbol{\epsilon}(\boldsymbol{x}) \boldsymbol{\epsilon}(\boldsymbol{x})^T) \boldsymbol{\zeta} \\ &= \boldsymbol{\zeta}^T \mathrm{Cov}(\boldsymbol{\epsilon}(\boldsymbol{x}), \boldsymbol{\epsilon}(\boldsymbol{x})) \boldsymbol{\zeta} \\ &= \sigma^2 \boldsymbol{\zeta}^T R \boldsymbol{\zeta} \end{aligned}$$

and since $E(\zeta^T \epsilon(\boldsymbol{x})\zeta^T \epsilon(\boldsymbol{x})) > 0$ if $\zeta \neq \mathbf{0}$ it follows that

$$\sigma^2 \zeta^T R \zeta > 0 \qquad \forall \zeta \in \mathbb{R}^n, \zeta \neq \mathbf{0} \tag{5.1}$$

and hence R is positive definite. It follows directly from the positive definiteness of R that R^{-1} is also positive definite.

We will now address the convexity issue in the problem of maximizing the mean squared error (MSE). Taken as a function in the variables $\boldsymbol{r}_x = (r_1, \ldots, r_n)$ and $\boldsymbol{x} = (x_1, \ldots, x_d)$, the objective function to be minimized is

$$\begin{aligned} -s^2(\hat{y}(\boldsymbol{x})) &= -\sigma^2[1 - \boldsymbol{r}_x^t R^{-1} \boldsymbol{r}_x + \boldsymbol{f}_x^t (F^t R F)^{-1} \boldsymbol{f}_x - 2\boldsymbol{r}_x^t R^{-1} F (F^t R F)^{-1} \boldsymbol{f}_x \\ &\quad + \boldsymbol{r}_x^t R^{-1} F (F^t R F)^{-1} F^t R^{-1} \boldsymbol{r}_x]. \end{aligned}$$

In the case where $\boldsymbol{f}_x = 1$ and $F = \mathbf{1}$ this simplifies to

$$-s^2(\hat{y}(\boldsymbol{x})) = -\sigma^2 \left[1 - \boldsymbol{r}_x^t R^{-1} \boldsymbol{r}_x + \frac{(1 - \mathbf{1} R^{-1} \boldsymbol{r}_x)^2}{\mathbf{1}^t R^{-1} \mathbf{1}}\right]. \tag{5.2}$$

In the simplified case with $\boldsymbol{f}_x = 1$ and $F = \mathbf{1}$ the function $-s^2(\hat{y}(\boldsymbol{x}))$ only depends on $\boldsymbol{r}_x$, not directly on $\boldsymbol{x}$, so the Hessian matrix H_{s^2} of the function $-s^2(\hat{y}(\boldsymbol{x}))$ is of the form

$$H_{s^2} = \begin{pmatrix} H_{rr} & 0 \\ 0 & 0 \end{pmatrix}.$$

where

$$H_{rr} = 2\sigma^2 \left[R^{-1} - \frac{R^{-1} \mathbf{1} \mathbf{1}^t R^{-1}}{\mathbf{1}^t R^{-1} \mathbf{1}}\right].$$

The eigenvalues of H_{s^2} are the eigenvalues of the upper left portion H_{rr} of the matrix H_{s^2}, and 0. Jones *et al.* [34] allow for the matrix H_{rr} to be indefinite, and αBB is used to solve the potentially non-convex optimization problem. We will show here that since R is positive definite, the matrix H_{rr} is positive semi-definite, that is

$$\boldsymbol{v}^t H_{rr} \boldsymbol{v} = 2\sigma^2 \boldsymbol{v}^t \left[R^{-1} - \frac{R^{-1} \mathbf{1} \mathbf{1}^t R^{-1}}{\mathbf{1}^t R^{-1} \mathbf{1}}\right] \boldsymbol{v} \geq 0 \qquad \forall \boldsymbol{v} \in \mathbb{R}^n. \tag{5.3}$$

The matrix R is positive definite, therefore non-singular Cholesky factors can be found, such that $R = LL^t$. Then

$$\begin{aligned} \boldsymbol{v}^t \left[R^{-1} - \frac{R^{-1} \mathbf{1} \mathbf{1}^t R^{-1}}{\mathbf{1}^t R^{-1} \mathbf{1}}\right] \boldsymbol{v} &= \boldsymbol{v}^t L^{-t} L^{-1} \boldsymbol{v} - \frac{\boldsymbol{v}^t L^{-t} L^{-1} \mathbf{1} \mathbf{1}^t L^{-t} L^{-1} \boldsymbol{v}}{\mathbf{1}^t R^{-1} \mathbf{1}} \\ &= (L^{-1}\boldsymbol{v})^t L^{-1} \boldsymbol{v} - \frac{((L^{-1}\boldsymbol{v})^t L^{-1} \mathbf{1})^2}{(L^{-1}\mathbf{1})^t L^{-1} \mathbf{1}}. \end{aligned}$$

By the Cauchy-Schwartz inequality

$$((L^{-1}\boldsymbol{v})^t L^{-1} \mathbf{1})^2 \leq (L^{-1}\boldsymbol{v})^t L^{-1} \boldsymbol{v} (L^{-1}\mathbf{1})^t L^{-1} \mathbf{1}$$

and therefore

$$(L^{-1}\boldsymbol{v})^t L^{-1}\boldsymbol{v} - \frac{((L^{-1}\boldsymbol{v})^t L^{-1}\mathbf{1})^2}{(L^{-1}\mathbf{1})^t L^{-1}\mathbf{1}} \geq 0.$$

Therefore

$$\boldsymbol{v}^t \left[R^{-1} - \frac{R^{-1}\mathbf{1}\mathbf{1}^t R^{-1}}{\mathbf{1}^t R^{-1}\mathbf{1}} \right] \boldsymbol{v} \geq 0 \qquad \forall \boldsymbol{v} \in \mathbb{R}^n$$

and H_{rr} and hence H_{s^2} is positive semi-definite.

The Hessian is generally not positive semi-definite if regression functions other than the constant one are used. In the case of a linear basis used for $\boldsymbol{f}_x$, i.e. $\boldsymbol{f}_x^t = (1, x_1, \ldots, x_d)$ for example, the Hessian of $-s^2$ becomes

$$H_{s^2} = 2\sigma^2 \begin{bmatrix} R^{-1} - R^{-1}F(F^tRF)^{-1}F^tR^{-1} & [R^{-1}F(F^tRF)^{-1}]_{-1} \\ ([R^{-1}F(F^tRF)^{-1}]_{-1})^t & -[(F^tRF)^{-1}]_{-1,-1} \end{bmatrix} \tag{5.4}$$

where $[R^{-1}F(F^tRF)^{-1}]_{-1}$ denotes the $n \times m$ matrix obtained by leaving out the first column of the matrix $R^{-1}F(F^tRF)^{-1}$, and similarly, $[(F^tRF)^{-1}]_{-1,-1}$ denotes the $m \times m$ matrix obtained by leaving out the first column and row of the matrix $(F^tRF)^{-1}$. Using Cholesky factors, the matrix R can be written as $R = LL^t$, where L is a lower-triangular matrix. Then applying Householder factorization to $L^{-1}F$ gives $L^{-1}F = Q^tU$, where Q is an orthogonal $n \times n$ matrix and U is an upper-triangular $n \times m$ matrix. U is of the form

$$U = \begin{pmatrix} U_1 \\ 0 \end{pmatrix}$$

where U_1 is an $m \times m$ upper triangular matrix. Let Q_1 be the $m \times n$ matrix such that

$$L^{-1}F = Q_1^t U_1.$$

Then we can rewrite the Hessian matrix as

$$H_{s^2} = \begin{bmatrix} 2\sigma^2[R^{-1} - L^{-t}Q_1^tQ_1L^{-1}] & 2\sigma^2[L^{-t}Q_1^tU_1^{-t}]_{-1} \\ (2\sigma^2[L^{-t}Q_1^tU_1^{-t}]_{-1})^t & -2\sigma^2[U_1^{-1}U_1^{-t}]_{-1,-1} \end{bmatrix}. \tag{5.5}$$

The matrix $U_1^{-1}U_1^{-t}$ is positive semidefinite, and so is $[U_1^{-1}U_1^{-t}]_{-1,-1}$. It follows that the matrix $-2\sigma^2[U_1^{-1}U_1^{-t}]_{-1,-1}$ is not positive semi-definite, and therefore the matrix H_{s^2} is not positive semi-definite.

We conclude that when, as is typically the case, a constant regression function is used, the Hessian matrix H_{s^2} of the negative mean squared error function $-s^2(\hat{y}(\boldsymbol{x}))$, which we are minimizing, is positive semi-definite and no convexification of this function is necessary. This is not generally true for example in the case of higher order polynomial regression functions, where the Hessian matrix H_{s^2} of the negative mean squared error function $-s^2(\hat{y}(\boldsymbol{x}))$ is not necessarily positive semi-definite.

5.3 Maximizing the Expected Improvement

From Chapter 3 we know that the expected improvement may be maximized using branch and bound, where an upper bound on the expected improvement EI_U on a region can be calculated from the maximum, $s_{\max}$, of the root MSE and the minimum, $\hat{y}_{\min}$, of the BLUP via

$$\mathrm{EI}_U = (y_{0n} - \hat{y}_{\min})\Phi\left(\frac{y_{0n} - \hat{y}_{\min}}{s_{\max}}\right) + s_{\max}\phi\left(\frac{y_{0n} - \hat{y}_{\min}}{s_{\max}}\right), \tag{5.6}$$

where y_{0n} is the best objective function value found at the first n sample points, and where $s_{\max}$ and $\hat{y}_{\min}$ are obtained by solving the following two subproblems Problem 1:

$$\begin{aligned} \min \quad & \hat{y}(\boldsymbol{x}, \boldsymbol{r}_x) = \boldsymbol{f}_x^t\hat{\boldsymbol{\beta}} + \boldsymbol{r}_x^t R^{-1}(\boldsymbol{y} - F\hat{\boldsymbol{\beta}}) \\ s.t. \quad & r_i - \prod_{j=1}^{d} \exp(-\theta_j |x_j - x_j^{(i)}|^{p_j}) = 0 \qquad i = 1, \ldots, n \\ & x_j^L \le x_j \le x_j^U \qquad j = 1, \ldots, d \\ & r_i^L \le r_i \le r_i^U \qquad i = 1, \ldots, n \end{aligned} \tag{5.7}$$

Problem 2:

$$\begin{aligned} \min \quad & -s^2(\boldsymbol{x}, \boldsymbol{r}_x) = -\sigma^2(1 - \boldsymbol{r}_x^t L^{-t} L^{-1} \boldsymbol{r}_x + \boldsymbol{f}_x^t U_1^{-1} U_1^{-t} \boldsymbol{f}_x \\ & \qquad -2\boldsymbol{r}_x^t L^{-t} Q_1 U_1^{-t} \boldsymbol{f}_x + \boldsymbol{r}_x^t L^{-t} Q_1^t Q_1 L^{-1} \boldsymbol{r}_x) \\ s.t. \quad & r_i - \prod_{j=1}^{d} \exp(-\theta_j |x_j - x_j^{(i)}|^{p_j}) = 0 \qquad i = 1, \ldots, n \\ & x_j^L \le x_j \le x_j^U \qquad j = 1, \ldots, d \\ & r_i^L \le r_i \le r_i^U \qquad i = 1, \ldots, n. \end{aligned} \tag{5.8}$$

Bounds on the expected improvement resulting from bounds on $\hat{y}$ and s^2 as solutions of (5.7) and (5.8) respectively, can be poor, as $\min \hat{y}$ and $\max s^2$ are typically not attained at the same sample point $\boldsymbol{x} \in B$, where B is the current working box. This is clear intuitively since $\min \hat{y}$ is typically at a point close to the best sample point and $\max s^2$ at the point furthest from any sample point. An example of this can be seen in Figures 5.1 and 5.2, which show the objective function and $\hat{y}$, and $\log(s^2)$ and $\log(\mathrm{EI})$, respectively. The minimum of the BLUP $\hat{y}$ is attained at a sample point (see Figure 5.1), where the MSE s^2 is zero (see Figure 5.2). Figure 5.2 also shows that the maximum expected improvement and the maximum of the MSE do not coincide. An example of the effect of this is shown in Figure 5.3. In this plot some contours of the expected improvement function are displayed, and also the locus of $(\hat{y}, s^2)$ is shown as x ranges between 0 and 1. For

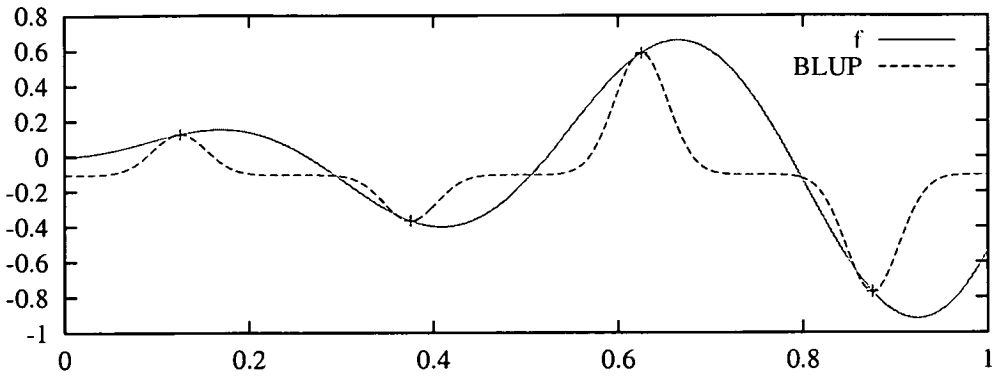

Figure 5.1: Objective function and best linear unbiased predictor $\hat{y}$

$x = 0$, s^2 takes a large value and $\hat{y}$ takes, essentially, a mean value. As x increases from 0 to 1, the locus touches $s^2 = 0$ at sample points, i.e. where $\hat{y}$ interpolates function values. Midway between sample points, and at the endpoints of $[0, 1]$, s^2 approaches its maximum value. The minimum value of $\hat{y}$ and the maximum value of s^2 are indicated by a dotted vertical and a horizontal line, respectively. The upper bound on the expected improvement obtained from $\min \hat{y}$ and $\max s^2$ is indicated where these two lines intersect in the top lefthand corner — this is $\mathrm{EI}_U = 0.2275$. The minimum value of $\hat{y}$ is the best actual function value found so far. The maximum expected improvement value of 0.0695 is attained at the point indicated, i.e. where the locus touches the EI = 0.0695 contour. As can be seen in Figure 5.1 and Figure 5.2, $\min \hat{y}$ and $\max s^2$ are not attained at the same point $\boldsymbol{x}$, and the upper bound on EI derived from $\min \hat{y}$ and $\max s^2$ is out by a factor of more than three.

The two problems (5.7) and (5.8) are non-convex problems by virtue of the nonlinear constraints. The linearizations of the nonlinear constraints as constructed by Jones *et al.* [34] and the fact that $-s^2(\boldsymbol{r})$ is a convex function, as shown earlier, mean that solving to find an upper bound on $\max s^2$ and a lower bound on $\min \hat{y}$ is a convex QP and LP in the case of constant regression functions being used. The solutions to this QP and LP give an upper bound s_U^2 on $\max s^2$ and a lower bound $\hat{y}_L$ on $\min \hat{y}$. However, the linearizations are not tight and therefore the bounds s_U^2 and $\hat{y}_L$ are poor. Coupled with the fact that, even without the linearizations of the constraints, the bounds on $\max s^2$ and $\min \hat{y}$ are not attained at the same point $\boldsymbol{x} \in B$, the linearizations lead to further weakness of the upper bound on the expected improvement, which is now obtained using

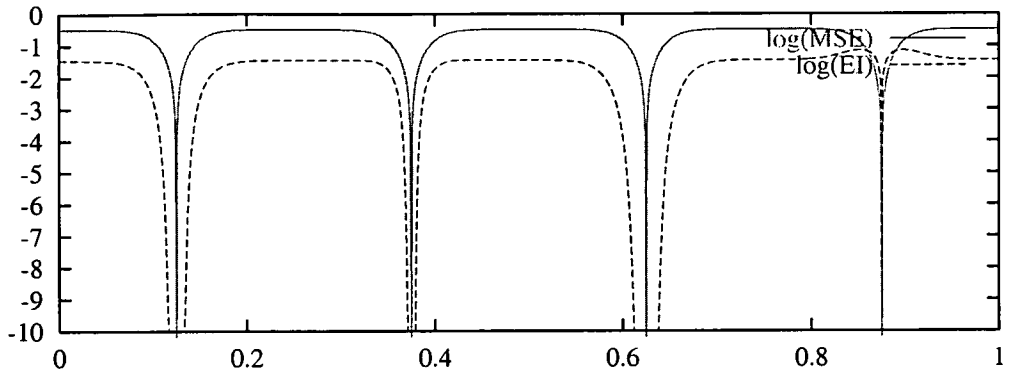

Figure 5.2: $\log(s^2)$ and $\log(\mathrm{EI})$

the bounds s_U on $\max s$ and $\hat{y}_L$ on $\min \hat{y}$, via

$$\mathrm{EI}_U = (y_{0n} - \hat{y}_L)\Phi\left(\frac{y_{0n} - \hat{y}_L}{s_U}\right) + s_U\phi\left(\frac{y_{0n} - \hat{y}_L}{s_U}\right).$$

An example of this is shown in Figure 5.4. Here the $\hat{y}$ and s^2 are the same as in Figure 5.3, but the linearization of the constraints has led to a worse lower bound $\hat{y}_L = -1.0191$ on $\hat{y}$ than before. Using this lower bound $\hat{y}_L$ and the upper bound $s_U^2 = 0.3253$ on s^2 to calculate an upper bound on the expected improvement gives $\mathrm{EI}_U = 0.3736$, compared with an upper bound on the expected improvement of $\mathrm{EI}_U = 0.2275$ in the previous figure, and an actual maximum expected improvement of $\mathrm{EI}_{\max} = 0.0695$. The bounds on $\hat{y}$ and s^2 do become tighter as boxes become much smaller in the process of the branch and bound, but much branching is required before boxes are small enough, and before this method gives bounds on $\hat{y}$ and s^2 which lead to useful upper bounds on the expected improvement. There are two issues here that can be addressed:

1. improvement of the bounding procedure for EI

2. improvement of the linearizations of the constraints.

An improvement of the bounding procedure for EI can be achieved by ensuring that it corresponds to values of $\hat{y}$ and s^2 which are near feasible, in terms of being attained at the same point $\boldsymbol{x} \in B$, or at two points close together in B.

In the following we will discuss these two issues, starting with the improved bounding for the expected improvement, then introducing improved linearizations of the constraints.

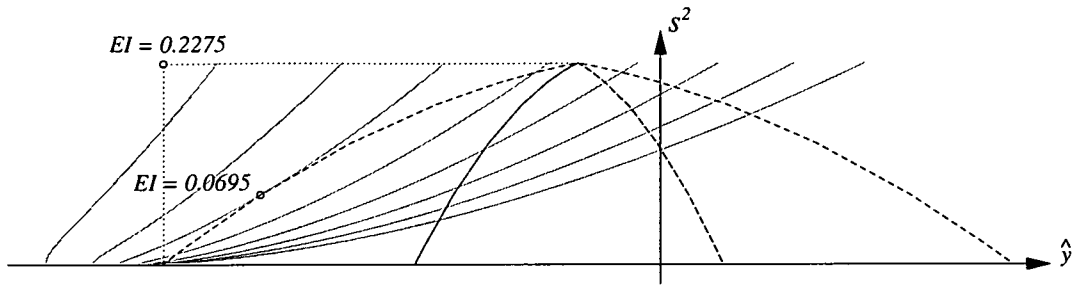

Figure 5.3: Upper bound on EI

5.3.1 Bound on EI

Consider the problem of maximizing

$$\mathrm{EI}(\hat{y}, s^2) = (y_{min} - \hat{y})\Phi\left(\frac{y_{min} - \hat{y}}{s}\right) + s\phi\left(\frac{y_{min} - \hat{y}}{s}\right) \tag{5.9}$$

as a function of $\hat{y}$ and s. For points $\boldsymbol{x}$ in a box B the points $[\,\hat{y}(\boldsymbol{x}) \quad s^2(\boldsymbol{x})\,]^t$ form a feasible region T and the problems

$$\begin{aligned} \max \quad & \mathrm{EI}(\boldsymbol{x}) \\ & \boldsymbol{x} \in B \end{aligned}$$

and

$$\begin{aligned} \max \quad & \mathrm{EI}(\hat{y}, s^2) \\ & [\,\hat{y}(\boldsymbol{x}) \quad s^2(\boldsymbol{x})\,]^t \in T \end{aligned}$$

are equivalent. The feasible region T is bounded by the lower bound $\hat{y}_L$ on $\hat{y}(\boldsymbol{x})$ and the upper bound s_U^2 on $s^2(\boldsymbol{x})$ for $\boldsymbol{x} \in B$. When linearizing the nonlinear constraints, more feasible points $[\,\hat{y}(\boldsymbol{x}) \quad s^2(\boldsymbol{x})\,]^t$ are introduced. Let the feasible set of points $[\,\hat{y}(\boldsymbol{x}) \quad s^2(\boldsymbol{x})\,]^t$ with respect to the linearized constraints be denoted by $\overline{T}$. The lower bound $\hat{y}_L$ and the upper bound s_U^2 on the set $\overline{T}$ are the bounds that were previously used to find an upper bound on EI by means of (5.9). Clearly, $s^2(\boldsymbol{x}) \geq 0$ and, by solving a further LP, an upper bound $\hat{y}_U$ can be found for $\hat{y}(\boldsymbol{x})$, so that we have upper and lower bounds on $s^2(\boldsymbol{x})$ and $\hat{y}(\boldsymbol{x})$. These bounds are illustrated in Figure 5.4. For any (fixed) constants $a < 0$ and $b > 0$, solving the concave QP problem

$$\begin{aligned} \max \quad & a\hat{y}(\boldsymbol{x}) + bs^2(\boldsymbol{x}) \\ & \boldsymbol{x} \in B \end{aligned} \tag{5.10}$$

subject to the linearizations of the nonlinear constraints yields a bound on T, since the line $a\hat{y} + bs^2 = c^*$ passing through the point $[\,\hat{y}^* \quad s^{2*}\,]^t$ at which the

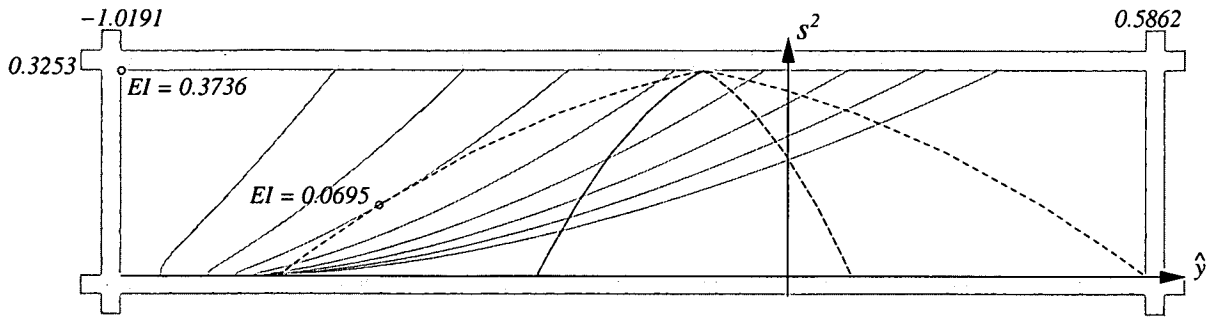

Figure 5.4: Upper bound on EI subject to linearized constraints

maximum of (5.10) is achieved, is either tangential to $\overline{T} \supset T$ or lies entirely outside $\overline{T}$.

Further, along any line $a\hat{y} + bs^2$ where $a < 0$, $b > 0$, it can be shown that $\mathrm{EI} \to 0$ as $\hat{y} \to \infty$. Although not yet proved, there is good reason to believe that along any such line $a\hat{y} + bs^2$ the expected improvement function only takes one local maximum, therefore a local optimization method would suffice to find $\max \mathrm{EI}$ along any such line. Then, (for any $a < 0, b > 0$)

$$\begin{array}{rcl} \mathrm{EI}(\hat{y}, s^2) & \leq & \max \mathrm{EI}(\hat{y}, s^2) \\ a\hat{y} + bs^2 \leq c^* & & a\hat{y} + bs^2 = c^* \end{array}$$

by the monotonicity properties of the expected improvement. Therefore

$$\begin{array}{rcl} \max \mathrm{EI} & \leq & \max \mathrm{EI}(\hat{y}, s^2) \\ \boldsymbol{x} \in B & & a\hat{y} + bs^2 = c^* \end{array}$$

so $\max \mathrm{EI}(\hat{y}, s^2)$ for $\hat{y}$ and s^2 on the line $a\hat{y} + bs^2 = c^*$ is an upper bound on the feasible $\max \mathrm{EI}$. Values for a and b are chosen heuristically, in an attempt to find a good upper bound on the expected improvement on $\overline{T}$. Figure 5.5 shows an example of the tighter upper bounds on the expected improvement. The bold straight lines are the lines $a\hat{y} + bs^2 = c^*$ for a sequence of values of a and b determined heuristically, and the maxima of the expected improvement along these lines are indicated. The fainter piecewise straight line is an approximation to the boundary of $\overline{T}$ obtained by solving (5.10) for a large number of values of a and b. The upper bound on the expected improvement found by the above method is $\mathrm{EI}_U = 0.1060$, compared with the upper bound as found by Jones *et al.* of $\mathrm{EI}_U = 0.3736$, the upper bound of 0.0882 on the (approximate) boundary of $\overline{T}$, and the actual maximum of the expected improvement $\mathrm{EI}_{\max} = 0.0695$. The work presented here on improving the expected improvement bounds is ongoing work together with Dr. Julian Hall.

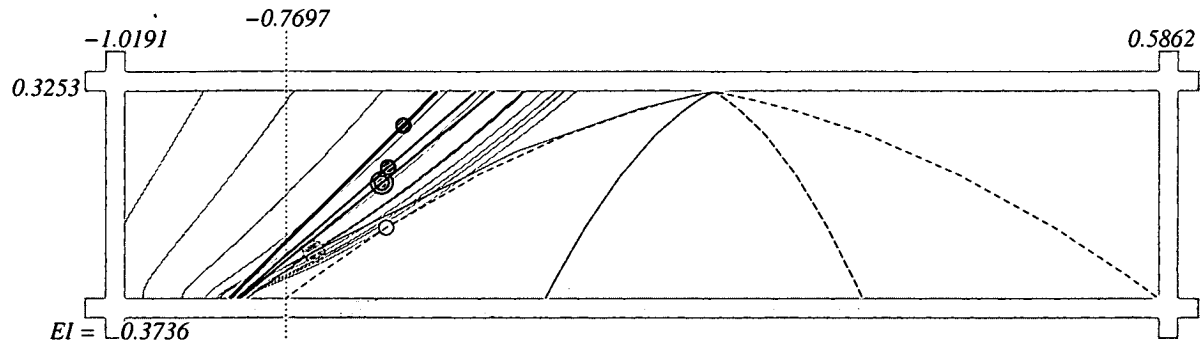

Figure 5.5: Improved upper bound on EI

5.3.2 Constraints

As we have seen, the dependence of $\boldsymbol{r}_x$ on $\boldsymbol{x}$ is determined by the constraints

$$r_i - \prod_{j=1}^{d} \exp(-\theta_j |x_j - x_j^{(i)}|^{p_j}) = 0 \qquad i = 1, \ldots, n. \tag{5.11}$$

As nonlinear equations, these constraints are non-convex. By relaxation of these constraints we can obtain a linearly constrained problem with a convex feasible region. Let us also recall that given lower bounds $\boldsymbol{x}_L$ and upper bounds $\boldsymbol{x}_U$ on $\boldsymbol{x}$ we can compute lower bounds $\boldsymbol{r}_L$ and upper bounds $\boldsymbol{r}_U$ on $\boldsymbol{r}_x$ using the fact that r_i depends on only one of the n sampling points, namely $\boldsymbol{x}^{(i)}$, and the value of r_i at any point only on the distance of that point to $\boldsymbol{x}^{(i)}$. As mentioned earlier, Jones *et al.* [34] split up the n equality constraints into $2n$ inequality constraints and replace them by linear underestimators. In the $2n$ inequality constraints

$$\ln(r_i) + \sum_{j=1}^{d} (\theta_j |x_j - x_j^{(i)}|^{p_j}) \leq 0 \qquad i = 1, \ldots, n \tag{5.12}$$

and

$$-\ln(r_i) - \sum_{j=1}^{d} (\theta_j |x_j - x_j^{(i)}|^{p_j}) \leq 0 \qquad i = 1, \ldots, n, \tag{5.13}$$

relaxing the first set of constraints (5.12) gives the overestimators and relaxing the second set of constraints (5.13) gives the underestimators of the actual constraints (5.11). Jones *et al.* relax the constraints by underestimating the terms involving r_i and $\boldsymbol{x}^{(i)}$ separately. The terms $\ln(r_i)$ for example are replaced by linear functions of the form $a + br_i$ which underestimate it over $[r_i^L, r_i^U]$. Similar underestimators are used to replace the other terms: in the case of $\ln(r_i)$ and $-|x_j - x_j^{(i)}|^{p_j}$ the underestimators are chords and in the case of $|x_j - x_j^{(i)}|^{p_j}$ and

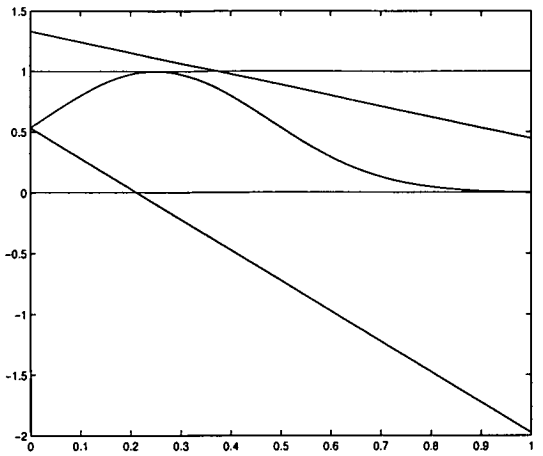

Figure 5.6: Constraints as found by Jones *et al.* in [34]

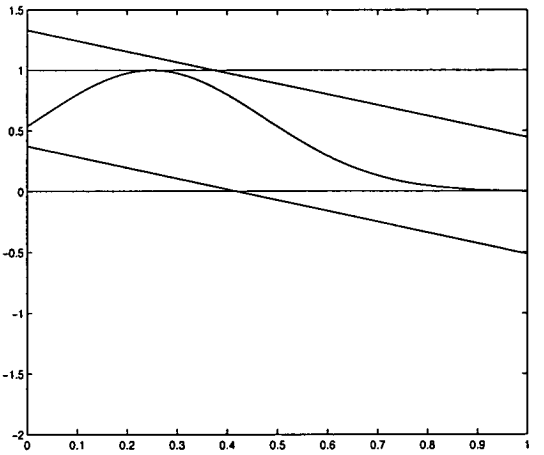

Figure 5.7: Lower constraint replaced by 'optimal' lower constraint

$-\ln(r_i)$ the underestimators are tangents. The tangents are computed at the midpoint of the relevant interval for x_j or r_i. It can be seen when plotting r_i and the linear over- and under-estimators, that when looking at larger boxes these linearizations are not very tight and typically are nowhere equal to r_i. As the branch and bound proceeds and the working boxes become smaller, these constraints do become tighter. Still, tighter constraints can be found, and by using a larger number of these tighter linear constraints, the feasible region can be made smaller. The limit of this is the convex hull of r_i on the relevant hyperrectangle $[\boldsymbol{x}_L, \boldsymbol{x}_U]$. One different underestimator is found easily. Rather than finding a tangent to $-\ln(r)$ at the midpoint of the box $[\boldsymbol{r}_L, \boldsymbol{r}_U]$ we can find a tangent that intersects the curve at an 'optimal' point. Denoting the tangent by $t(r)$ we can write our condition for this 'optimal' tangent as

$$-\ln(r_L) - t(r_L) = -\ln(r_U) - t(r_U),$$

i.e. the difference between the function and the tangent should be the same at both ends of the interval. This condition ensures that the maximum error in the linearization is minimized. Using $t(r) = -\frac{1}{r_t}(r - r_t) - \ln(r_t)$ as the equation of the tangent that intersects the curve in the point r_t we find

$$r_t = \frac{r_L - r_U}{\ln(r_L) - \ln(r_U)}$$

to be the desired point at which to construct the tangent. Note that the tangent to $\ln(r)$ which intersects the curve at r_t has slope

$$m_t = \frac{\ln(r_L) - \ln(r_U)}{r_L - r_U}$$

and is therefore parallel to the chord to $\ln(r)$ through the points $(r_L, \ln(r_L))$ and $(r_U, \ln(r_U))$. Figures 5.6 and 5.7 show the function $r_i = \exp(-\theta(x - x^{(i)})^p)$ for

$x^{(i)} = 0.25$, $\theta = 10$, and $p = 2$. Figure 5.6 shows upper and lower bounds on r_i and linear over- and underestimators on the box $[0, 1]$, as found by Jones *et al.* in [34]. For comparison, Figure 5.7 shows the function and the same bounds and linear overestimators as before, but the 'optimal' underestimators for the function on the box $[0, 1]$.

Other, tighter constraints can be constructed if, instead of bounding the terms $\ln(r_i)$ and $-|x_j - x_j^{(i)}|^{p_j}$ separately, upper and lower estimates are found for $r_i = \prod \exp(-\theta_k(x_k - x_k^{(i)})^{p_k})$. Figure 5.8 shows the function $r_i = \exp(-\theta(x - x^{(i)})^p)$ for $x^{(i)} = 0.25$, $\theta = 10$, and $p = 2$, the bounds on r_i, and two upper and two lower constraints. These constraints are tighter than the ones shown in the previous plots. In this particular case the two lower estimates are tangents: one is a tangent to $r(x)$ at the bound of the box which is further away from $x^{(i)} = 0.25$, and the other is a tangent which intersects the curve at the nearer bound relative to the point $x^{(i)}$. The upper estimates here are both tangents. Depending on the position of the point $x^{(i)}$ relative to the lower bounds $\boldsymbol{x}_L$ and upper bounds $\boldsymbol{x}_U$ of the box, these upper and lower estimates can be tangents or chords. It is clear that the function $r_i = \prod \exp(-\theta_k(x_k - x_k^{(i)})^2)$ attains its maximum at the point $x^{(i)}$, and if $x^{(i)}$ is not in the interval only one upper constraint is necessary. Other points of interest are the points of inflection of the function r_i, which are $x^{(i)} \pm \frac{1}{\sqrt{2\theta}}$. If the point of inflection on one side of the maximum is not in the current interval, but the maximum is, then it is clear that the linear overestimator on that side has to be a tangent. If neither point of inflection is in the interval, the one linear underestimator will be a chord. Over- and underestimators can be constructed in a similar way in d dimensions.

As the algorithm progresses new sampling points will be added close to previous sampling points, so that at some point we have say $x^{(i)}$ and $x^{(j)}$, $1 \leq i, j \leq n$ such that $x^{(i)} - x^{(j)} \leq \delta$. The difference of the two correlation functions $r_i(x)$ and $r_j(x)$,

$$r_i(x) - r_j(x) = \exp(-\theta(x - x^{(i)})^2) - \exp(-\theta(x - x^{(j)})^2)$$

will only take small values. Note that this is also the case if θ is small. Bounding this difference of the two correlation functions leads to a further reduction of the size of the feasible region of the relaxed problem. In one dimension we use two overestimators and two underestimators for these cuts on the differences. With a general way of obtaining the overestimators, the underestimators can be obtained easily by symmetry properties of the difference function about the point $\frac{x^{(i)} - x^{(j)}}{2}$. Figure 5.9 shows an example of such difference cuts obtained from one of our test functions, starting with an initial design of size $n = 8$ and adding the

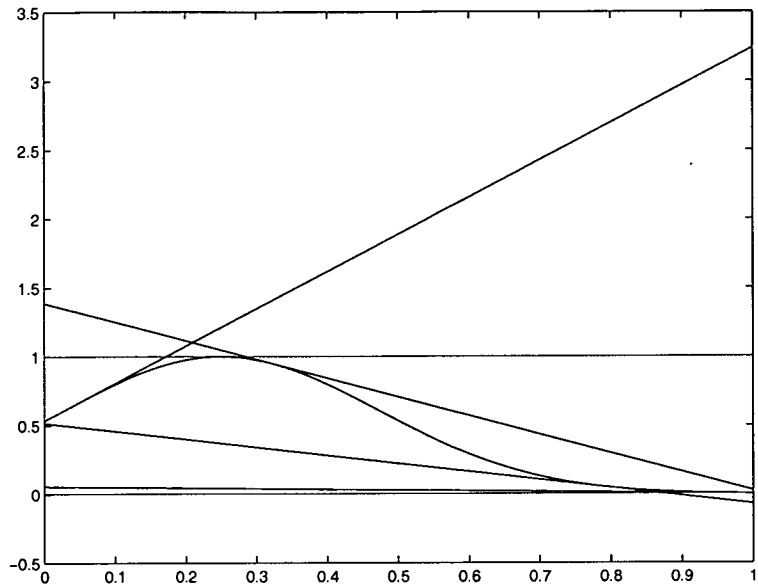

Figure 5.8: correlation function with tighter constraints

first set of cuts, as shown, after the next sample point is added. It is clear from Figure 5.9 that it is useful to know the position of the maximum, or minimum, of the function $r_i(x) - r_j(x)$ to be able to decide which cuts to use. We can calculate these using a little trick. Essentially the function $r_i(x) - r_j(x)$ depends only on the distance between the two points $x^{(i)}$ and $x^{(j)}$. So we can, instead of the function

$$r_i(x) - r_j(x) = \exp(-\theta(x - x^{(i)})^2) - \exp(-\theta(x - x^{(j)})^2),$$

consider the function

$$r_\delta(x) = \exp(-\theta(x - \delta)^2) - \exp(-\theta x^2).$$

Then we have

$$\begin{aligned} r_\delta(x) &= \exp(-\theta(x-\delta)^2) - \exp(-\theta x^2) \\ \frac{dr_\delta}{dx} &= -2\theta(x-\delta)\exp(-\theta(x-\delta)^2) + 2\theta x \exp(-\theta x^2) \\ \frac{d^2 r_\delta}{dx^2} &= (-2\theta + 4\theta^2(x-\delta)^2)\exp\left(-\theta(x-\delta)^2\right) + (2\theta - 4\theta^2 x^2)\exp\left(-\theta x^2\right) \end{aligned}$$

and using the Taylor expansion T'_δ of $\frac{dr_\delta}{dx}$ we can get a good approximation to the point where $\frac{dr_\delta}{dx}$ is zero. Substituting

$$\hat{x} = a_0 + a_1\delta$$

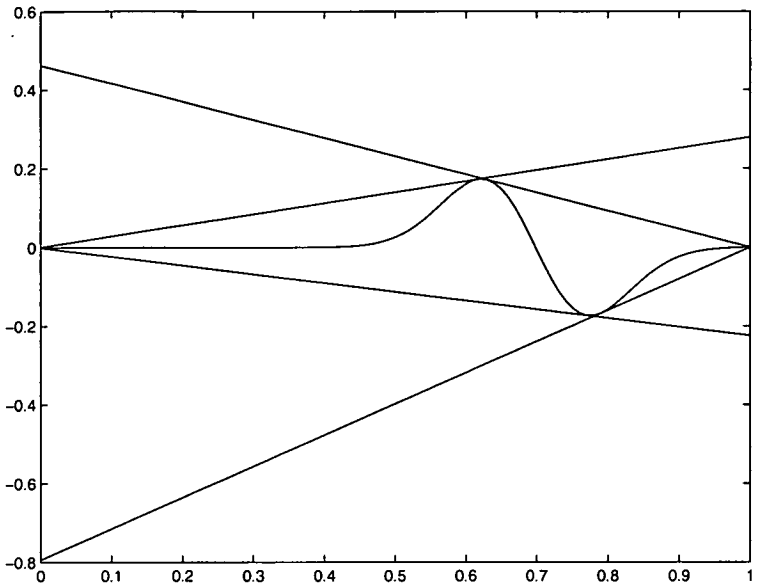

Figure 5.9: Difference cuts

into T'_δ gives

$$\frac{dr_\delta}{dx}(\hat{x}) = (-2\theta \exp(-\theta a_0^2)(2\theta a_0^2 - 1))\delta + O(\delta^2).$$

The coefficient of the δ term in this series depends only on a_0 and we can solve $T'_\delta = 0$ to find the coefficient

$$a_0 = \frac{1}{\sqrt{2\theta}}.$$

Then setting

$$\hat{x} = \frac{1}{\sqrt{2\theta}} + a_1\delta + a_2\delta^2$$

and substituting $\hat{x}$ into T'_δ gives

$$\frac{dr_\delta}{dx}(\hat{x}) = \left(-2\theta^{3/2} \exp\left(-\frac{1}{2}\right) 2^{1/2}(2a_1 - 1)\right)\delta^2 + O(\delta^3)$$

and we can solve $T'_\delta = 0$ to find the coefficient

$$a_1 = \frac{1}{2}.$$

This process can be continued in this manner, and gives a good approximation $\hat{x}$ to the stationary point of r_δ. This could be further refined for example with a few Newton-Raphson iterations starting with $x_1 = \hat{x}$.

When working to construct the overestimators or underestimators, knowing the position of the maximum helps to decide which overestimator to use. To the

x	$y(x)$	max EI	number of boxes in branch and bound best	Jones	new	Jones, cut	new, cut
0.0676	-1.045935	0.10270516	99	101	99	101	99
0.	-0.039352	0.02478984	65	93	79	79	77
0.1722	-0.700189	0.01318763	109	147	145	115	111
1.0000	0.091607	0.01287883	33	93	83	57	53
0.0764	-1.061679	0.00726441	33	105	93	71	71
0.5147		0.00099238	77	251	181	133	125

Table 5.1: Numbers of boxes used in branch and bound for different constraints

left of the maximum of the difference function the overestimator is a tangent or chord with positive slope, to the right of the maximum the overestimator is a tangent or chord with negative slope. In particular, if the maximum is not in the interval, only one overestimator is required.

The tightening of the constraints can lead to tighter bounds in the branch and bound and can help to reduce the number of boxes used. An example of this is shown in Table 5.1 where branch and bound and different types of constraints were used to solve the same problem, all with 6 initial points. Displayed are the sample points found, the function value, the maximum expected improvement, and the numbers of boxes that were used in the branch and bound to find the maximum expected improvement, and therefore the new sample point. The last row shows the iteration where the stopping tolerance of 0.001 on the expected improvement is met, and the objective function is not evaluated anymore at the point shown. The limit to what can be done by just tightening the linearizations of the constraints, is to satisfy the constraints exactly. The number of boxes in the maximization of the expected improvement based on this is shown under "best" in the table. It should be noted that this approach requires exhaustive search here and is not practical. Compared with it are the linearized constraints as done by Jones *et al.* in [34], the new linearizations as shown in 5.8, and both of these kinds of constraints with cuts, respectively. For the maximization of $s^2(\boldsymbol{x})$ and the minimization of $\hat{y}(\boldsymbol{x})$ an optimization routine from the Harwell Subroutine Library is used, see for example [29].

A combination of improved calculation of the bounds on the expected improvement and tighter constraints could bring more improvement, but this has yet to be investigated.

5.4 Grid Search versus Branch and Bound

As discussed in the previous section, the bounds on the expected improvement obtained for the branch and bound are typically not tight and much branching is required before these bounds become useful and some boxes can be discarded. Two obvious solutions to this problem are (1) improving the bounding in branch and bound and (2) not to use branch and bound at all. The possibility of the former of these two solutions, improving the branch and bound, was investigated in previous sections, where it was shown how the bounding of the expected improvement can be improved and how the linearizations of the constraints can be tightened. The best that can possibly be done by tightening the relaxations of the constraints is to find the actual $s_{\max}$ and $\hat{y}_{\min}$, i.e. satisfying the constraints exactly, and calculate the upper bound for the expected improvement via (5.6). Even then a considerable number of boxes is required, as can be seen in Table 5.2. Here the six initial sample points are shown and then the new sample points for the following iterations, along with their objective function value, maximum expected improvement and the number of boxes used to find the maximum expected improvement. The last maximum expected improvement displayed in the table is smaller than the tolerance of 0.001 and the algorithm stops. Combining better bounding with tighter constraints will improve the branch and bound even further than applying each of these two improvements separately, but it is not clear to what extent it leads to a reduction in the number of boxes used. If the branch and bound is prohibitively expensive then one should resort to solution (2) and consider another optimization method as a compromise, for example by using a grid search, for optimizing the expected improvement. Besides investigating the branch and bound method for maximizing the expected improvement function, we use a grid search combined with a local search method. The expected improvement is evaluated at a number of grid points and, optionally, a local search can be started from the grid point with the best function value. The resulting point is used as a new sampling point.

Notably, while Jones *et al.* in [34] use branch and bound to maximize the expected improvement to global optimality, the same group of people later state in [64], quoting Mockus in [47], that maximizing the expected improvement to global optimality is not necessary, since it is only used to determine the point of the next observation. In our experience, if a local maximizer of the expected improvement is used as the new sampling point, other local maxima, and particularly the global maximum of the expected improvement function are more or less preserved in the next iteration and it is likely that, even if we only search for a local maximum,

x	$y(x)$	max EI	boxes
0.5833	-0.011582		
0.7500	0.799848		
0.9167	1.120453		
0.4167	0.653336		
0.2500	-2.274116		
0.0833	-1.049753		
0.2042	-2.348656	0.18000015	67
0.2213	-2.439180	0.03591238	27
0.	-0.126714	0.00530989	31
0.2245	-2.440447	0.00248054	31
0.2237		0.00015942	26

Table 5.2: Data points, maximum expected improvement and number of boxes used in branch and bound

we will eventually find the global one and use the point as a new sample point. If we do choose the global maximizer as a new sampling point it is likely that the algorithm still explores the other local maxima in the following iterations anyway. Solving the maximization problem only to local optimality is easier and cheaper to do.

An issue is that a global method in some iteration could tip the balance between finding the global maximum of the expected improvement and continuing, because the maximum is higher than the stopping tolerance, and stopping because the global maximum was missed and the stopping tolerance satisfied. Table 5.2 and Figure 5.10 shows an example where this could be the case. Plotted in Figure 5.10 are the objective function as a solid line, the BLUP as a dashed line, and the expected improvement, scaled to make it more pronounced. Indicated are also the global minimum of the objective function by a + and the sample points by dots. The actual maximum expected improvement value here is $\mathrm{EI}_{\max} = 0.00248054$. Continuing to a stopping tolerance of 0.001 on the maximum expected improvement, the objective function would be evaluated and the algorithm would continue. Maximizing by another method, it is not unlikely that this peak in the expected improvement would have been missed, and the algorithm would have stopped. In such a situation maximizing the expected improvement to global optimality can lead to a more accurate solution of the minimization of the objective function. On the other hand, insisting on sampling at global maxima only of the expected improvement function does certainly not guarantee successful optimization of the objective function. The expense of branch and bound and the fact that the usefulness of maximizing the expected improvement to global optimality (except maybe in the potentially final iteration) is doubtful, leads us to think

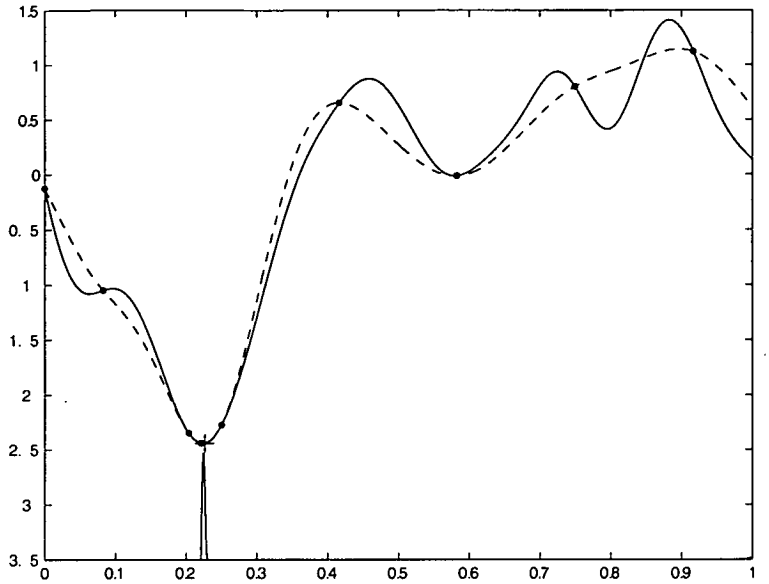

Figure 5.10: Objective function, BLUP, and EI

that, unless the method of globally maximizing the expected improvement can be improved significantly, local optimization methods are preferable.

5.5 Parameter Estimation

Sometimes the algorithm fails to locate the global optimum of a function, and occasionally it even fails to locate a local minimum. One reason for this can be that in the parameter estimation the d parameters $\boldsymbol{\theta}$ are underestimated, and the BLUP becomes too smooth and the MSE becomes too small. This causes a problem if the termination condition is met in this iteration. If it is not met and the algorithm carries on at the next sample point $\boldsymbol{x}$ the newly found objective function value $y(\boldsymbol{x})$ is unlikely to match the value of the predictor $\hat{y}(\boldsymbol{x})$ well, and $\boldsymbol{\theta}$ in the following iteration will be estimated to be something more appropriate for the shape of the actual function. However, this is not always the case: if at the next sample point $\boldsymbol{x}$ the newly found objective function value $y(\boldsymbol{x})$ matches the value of the predictor $\hat{y}(\boldsymbol{x})$ well, then $\boldsymbol{\theta}$ is again underestimated in the following iteration. A drastic example of this is shown in Figure 5.11. Displayed are the objective function, a sample path from a process with $\theta = 225$, $\mu = 0$, and $\sigma = 1$, the BLUP (dashed), and at the bottom of the plot the expected improvement function (shifted down). The parameters are estimated to be $\hat{\theta} \approx 6.80645$, $\hat{\mu} \approx 0.05571$, and $\hat{\sigma} \approx 1.43520$. One sample point is added, $x^{(7)} = 0$, at which the objective function and the BLUP are in close agreement. The new parameter estimates are $\hat{\theta} \approx 6.12500$, $\hat{\mu} \approx 0.20861$, $\hat{\sigma} \approx 1.47324$, and

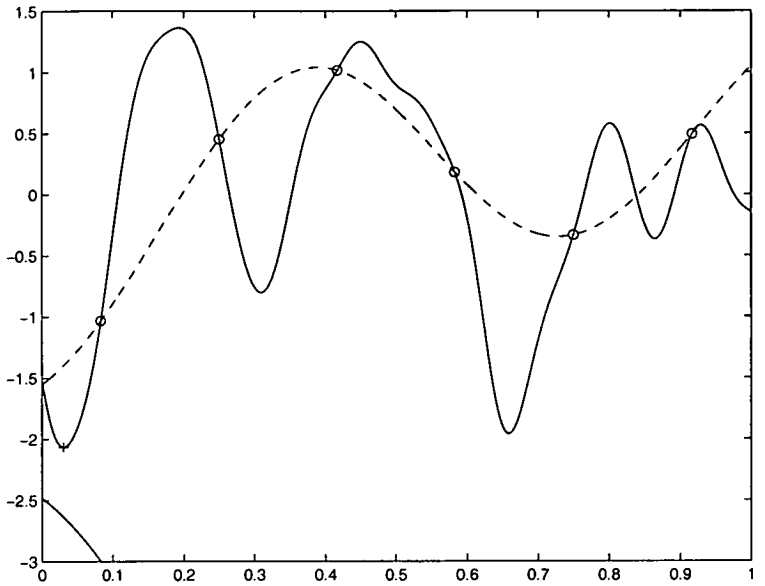

Figure 5.11: Objective function, BLUP, and EI at first iteration for Problem 48, generated with $\theta = 225$, 6 initial sample points

the maximum expected improvement is very small, so the algorithm stops. The best point found is $\boldsymbol{x}^{(7)}$, which is not even a local minimizer, and its objective function value is more than 0.5 above the global minimum. This example is an extreme failure case which is difficult to fix and might just have to be put down as bad luck, it is the worst failure case found in the 500 problems in T225, but it is a good example of how a problem solve can go wrong.

Problems can also occur if the objective function has, besides the global optimum, a good local optimum, and if the initial design has sampled points with good function values near the local optimum but not near the global one. It can then be the case that $\boldsymbol{\theta}$ is underestimated, so we expect high correlation and less variation in the function values, therefore keep sampling around the local optimum and miss the global one. To spot this kind of problem our generated test functions as described in Chapter 4 are useful. The estimated parameters $\hat{\boldsymbol{\theta}}$ can be expected to be close to the values which we used to generate the objective function, and we can directly compare these values. The estimates $\hat{\boldsymbol{\theta}}$ are obtained through maximum likelihood estimation and often the log likelihood function is very flat around the maximizer $\boldsymbol{\theta}^*$ such that if we increase $\hat{\boldsymbol{\theta}}$ away from $\boldsymbol{\theta}^*$ and the origin, the log likelihood at $\hat{\boldsymbol{\theta}}$ and $\boldsymbol{\theta}^*$ differs only by a small amount. That means that larger values for $\hat{\boldsymbol{\theta}}$ are often almost equally likely as $\boldsymbol{\theta}^*$. This suggests that larger $\hat{\boldsymbol{\theta}}$ matches our data almost equally well, and in fact an increase in $\hat{\boldsymbol{\theta}}$ can sometimes mean a more successful run of the algorithm. Figure 5.12 shows the stopping situation in an example where the BLUP for the 12 initial sample

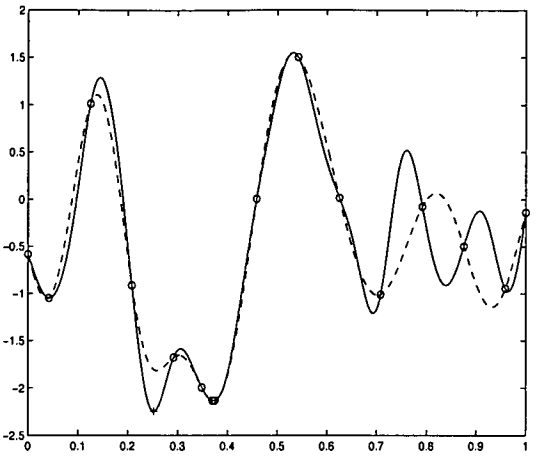

Figure 5.12: Final situation in the optimization of problem 24 generated with $\theta = 225$, 12 initial points, θ estimated by maximum likelihood

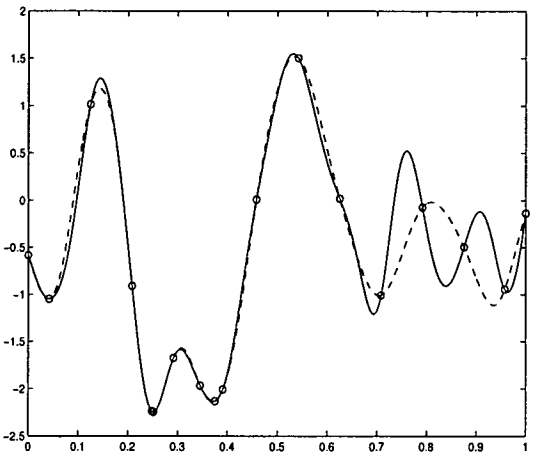

Figure 5.13: Final situation in the optimization of problem 24 generated with $\theta = 225$, 12 initial points, θ fixed to 225

points captures a good local optimum, but misses the nearby global optimum. Displayed are the BLUP (dashed), the objective function, the sample points and the global optimum. Had θ been known to be 225 and set to that value throughout the optimization process, the stopping situation would have been as in Figure 5.13 and the global optimum of the objective function would have been found.

Another problem, related to the underestimation of $\boldsymbol{\theta}$, seems to be the underestimation of σ^2. We have observed a number of cases in our generated functions where σ^2 is underestimated and the algorithm does not allow for functions with more variation from the predictor. An example of this is shown in Figure 5.14. The objective function is generated from $\theta = 225$, $\mu = 0$, and $\sigma = 1$. The maximum likelihood estimates of the parameters in this particular iteration are $\hat{\theta} \approx 255.8$, $\hat{\mu} \approx 0.065358$, and $\hat{\sigma} \approx 0.571866$. The algorithm terminates here having missed the global minimum by more that 0.8. Figure 5.15 shows the final state if the value of σ had been known to be 1 and this value had been used throughout the optimization, with the maximum likelihood estimates used for θ and μ only. In this case the algorithm terminates correctly having found the global minimum.

To summarize, in most cases of observed difficulties or failures in the global optimization, the $\boldsymbol{\theta}$ or σ^2 were underestimated and the BLUP became too smooth or the MSE too low. Can this be taken into account in the expected improvement criterion? This question and possible answers will be investigated later in this chapter.

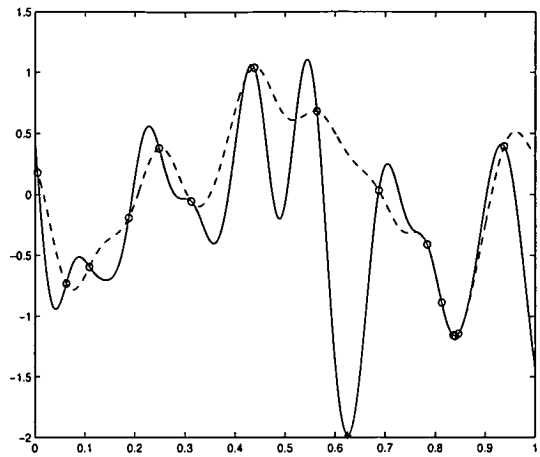

Figure 5.14: Final situation in the optimization of problem 20 generated with $\theta = 225$, 8 initial points, σ estimated by maximum likelihood

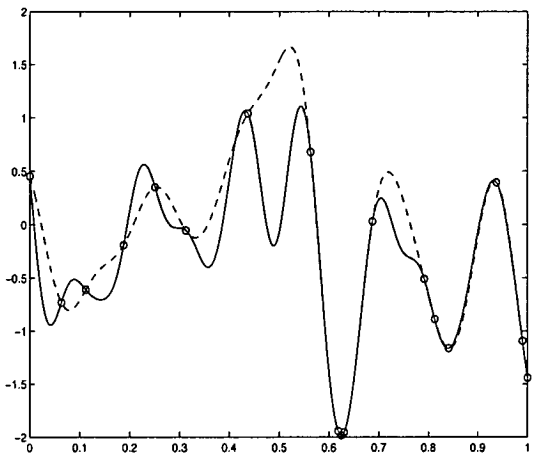

Figure 5.15: Final situation in the optimization of problem 20 generated with $\theta = 225$, 8 initial points, σ fixed to 1

5.6 Stopping Criterion and Number of Initial Points

The performance of the algorithm depends very much on the stopping criterion used. In every iteration Jones *et al.* in [34] compare the expected improvement with the current best function value, the criterion is to stop when the maximum expected improvement at a point $\boldsymbol{x} \in D$ is less than 1% of the absolute value of the current best function value. We have experimented with a stopping criterion derived from this. If the current best function value is close to zero the stopping tolerance used by Jones *et al.* can be very small. We therefore use two values to specify the stopping criterion and stop when the expected improvement is less than the maximum of a tolerance t_1 as used by Jones *et al.* and another absolute tolerance t_2. So the algorithm stops if

$$\max_{\boldsymbol{x} \in [0,1]^d} \mathrm{EI}(\boldsymbol{x}) \leq \max(|y_{0n} t_1|, t_2). \tag{5.14}$$

For example Jones *et al.* suggest using $t_1 = 10^{-2}$ and for t_2 we usually use $t_2 = 0.5 t_1$. Figure 5.16 shows the total average number of sample points (i.e. function evaluations) for 500 1-dimensional test problems generated as described in Chapter 4, with $\theta = 225$, $\mu = 0$ and $\sigma^2 = 1$, plotted against the average final maximum improvement (i.e. the difference between the global minimum and the best point found) for different numbers of starting points $2, 4, \ldots, 20$. Each curve corresponds to runs with a particular number of starting points, and the nine points on the curve correspond to nine different stopping tolerances used, with the lines only serving the purpose of connecting these points. The values used

for t_1 are $t_1 = 10^{-5}10^{i/4}$, $i = 0, \ldots, 8$ and the stopping rule used is as in (5.14). An observation can be made regarding the number of initial sample points here: the use of 18 or 20 initial sample points, for example, for this set of test functions T225 with $\theta = 225$, while pushing up the average of the total number of sample points, does not improve the performance significantly.

The actual stopping tolerance for the expected improvement depends on the value y_{0n} here, and it is difficult with this relative criterion to draw any conclusions regarding performance. We therefore decided to use an absolute stopping tolerance, independent of any function values, and stop whenever

$$\max_{\boldsymbol{x} \in [0,1]^d} \mathrm{EI}(\boldsymbol{x}) \leq t$$

for some tolerance t. Figure 5.17 shows a similar plot to Figure 5.16, but for the absolute stopping tolerance. For a fixed stopping tolerance on the expected improvement, the average total number of sample points (i.e. function evaluations) and the average final maximum improvement (i.e. the difference between the global minimum and the best function value at a sample point) are found. Shown in Figure 5.17 are the total average number of sample points for the set of test problems T225, plotted against the average final maximum improvement on the x-axis, for a range of different stopping tolerances on the expected improvement. Most results will be presented in this way, since what we are interested in is to achieve a small final error with few sample points. For this family of problems the best number of starting points depends on the required average final error. Figure 5.18 shows the results for the set of test problems T025. For this family of problems 4 starting points is optimal for the entire range of required final errors. It becomes clear from these pictures that we cannot make a single recommendation for a sensible number of starting points, as the performance depends strongly on the values of θ and on the required accuracy. Figure 5.18 shows that for these functions with $\theta = 25$, the use of 10 starting points increases the total number of function evaluations but does not improve performance significantly. Figure 5.19 shows the results for the set of test problems Txxx.

If the algorithm stops because the expected improvement satisfies the stopping criterion, the global optimization run counts as 'proper' completion of the algorithm. The algorithm can be expected to produce more accurate results for a smaller stopping tolerance, but using a very small stopping tolerance makes it more likely that the problem becomes near-singular in the process of the algorithm.

The expected improvement criterion often seems to underestimate the actual possible improvement and the algorithm stops prematurely, missing the global

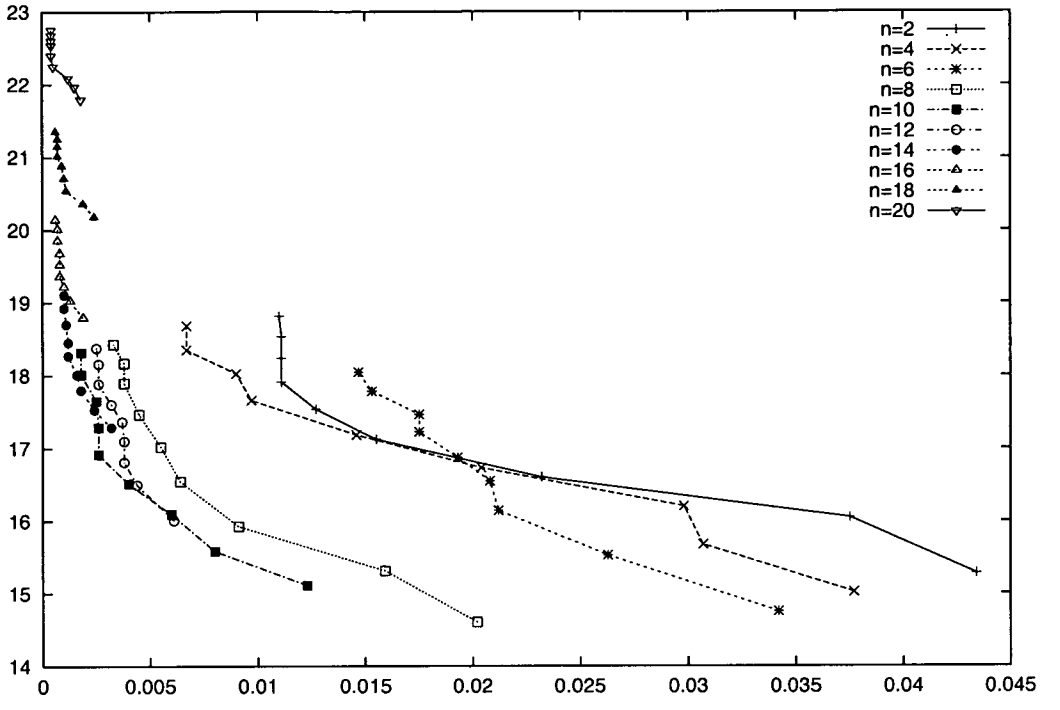

Figure 5.16: Average total number of points and average value difference for stopping tolerance depending on y_{0n}, test set T225

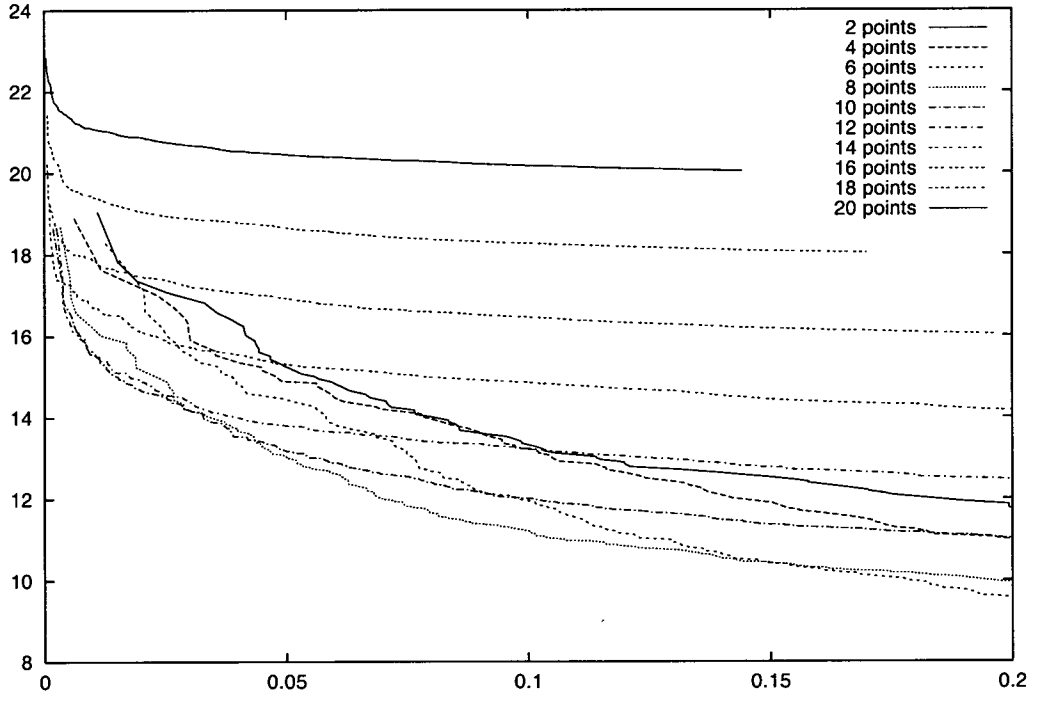

Figure 5.17: Average total number of points and average value difference for absolute stopping tolerance, test set T225

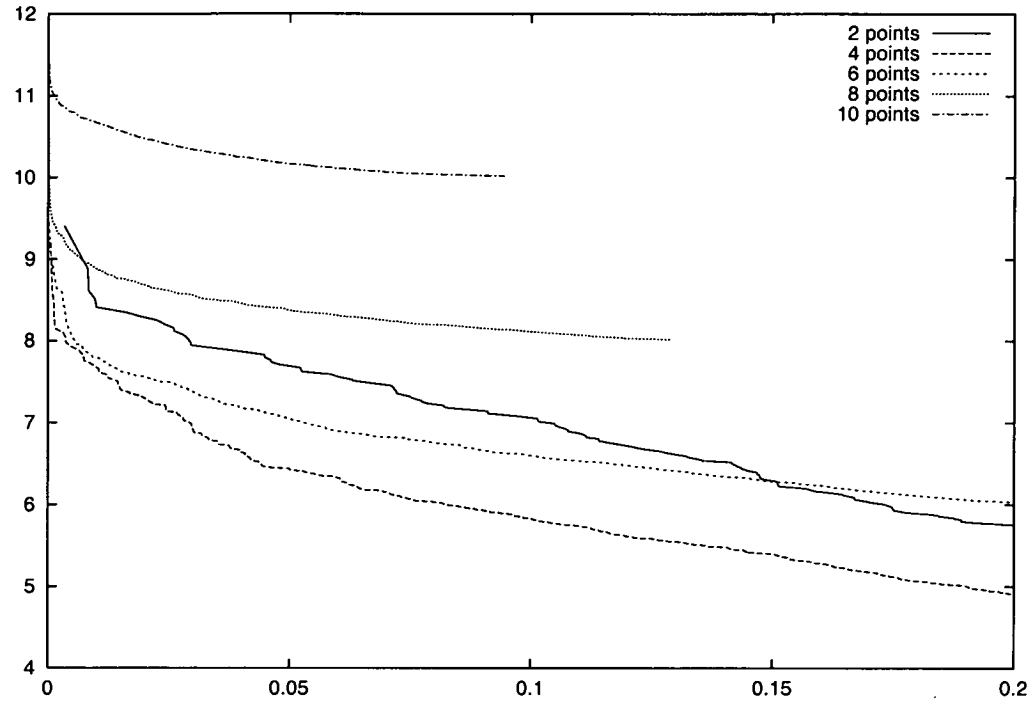

Figure 5.18: Average total number of points and average value difference for absolute stopping tolerance, test set T025

optimum in some cases. From Figure 4.20 in Chapter 4 it became clear that, had the uncertainty in the estimated parameters been taken into account, the optimization process would have continued and quite likely the global optimum of the objective function would have been found. Is it possible to incorporate this into a stopping rule? In the following we will investigate the possibility of expected improvement variants based on this idea.

5.7 Expected Improvement Variants

As the examples in Section 5.5 have demonstrated, in some cases the expected improvement criterion based on the maximum likelihood estimates of the parameters, fails to pick up the true improvement in the objective function and the algorithm stops prematurely. This often happens because the parameters $\boldsymbol{\theta}$ or σ, and therefore the variation in the objective function, are underestimated. Is it possible to take the uncertainty in the estimates of the parameters into account in the expected improvement criterion? The aim is to pick up previously missed global minima of the test functions while not increasing the number of objective function evaluations in successful runs. Another difficulty is that the choice of the number of initial points for a problem solve is not straightforward, but the

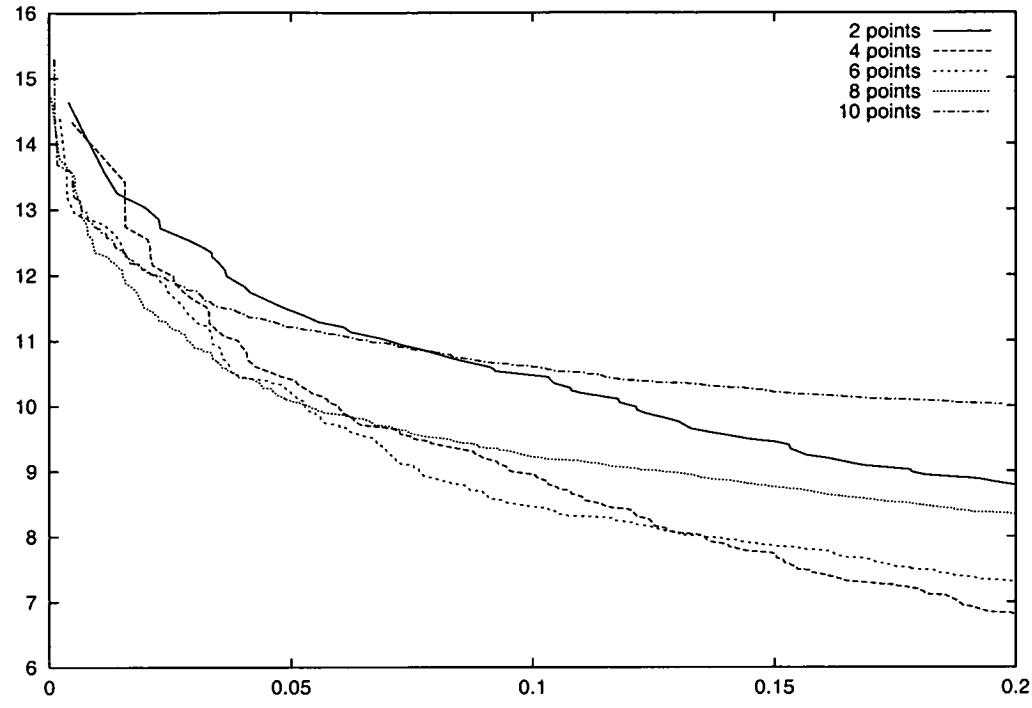

Figure 5.19: Average total number of points and average value difference for absolute stopping tolerance, test set Txxx

effectiveness and success of the solve depends on the number of starting points used. Is there a possibility that with another sampling criterion this choice would be less crucial, and a small number of starting points would suffice?

One possible solution is to take into account a measure of the width of the likelihood function and to vary the parameters accordingly. Another possibility is to use a Bayesian approach to find the posterior distribution of the parameters and use this, rather than the single point estimate produced by maximum likelihood, to calculate the expected improvement. One way to find the expected improvement corresponding to the posterior distribution of parameters is to use Monte Carlo sampling to generate alternative values for the parameters. For each combination of parameters generated we can then generate a conditional function passing through the sample points using the technique described in Section 4.3, and by averaging these get an estimate of the expected improvement. These issues motivate the experiments presented in the remainder of this chapter.

5.7.1 Varying Parameters

In this subsection we describe a method which calculates several different expected improvement functions using different estimates of the parameters of the function.

The different parameter values are chosen to be other likely values, and in choosing them we take into account the width of the likelihood function. A weighted average of these expected improvement functions can then be used as a sampling and stopping criterion. The estimate of the width of the likelihood function, in terms of $\boldsymbol{\theta}$ that we use, is the change in $\boldsymbol{\theta}$ needed to produce a drop in a local approximation of the likelihood function to 60% of the maximum likelihood value. For fixed $\boldsymbol{\theta}$ the maximum likelihood estimates of μ and σ, can be found. These are $\hat{\mu}$ and $\hat{\sigma}$ respectively. Then the deviation in σ is approximated by $\frac{n}{n-1}\sqrt{\frac{2}{n-1}}\hat{\sigma}^2$, and the deviation in terms of μ is taken to be $\frac{\hat{\sigma}}{\sqrt{\mathbf{1}^t\mathbf{R}^{-1}\mathbf{1}}}$. Taking combinations of parameters, increased or decreased by multiples of the estimates of their deviation, takes into account the uncertainty in the parameters. Sampling is initially done as before, by maximizing the expected improvement function based on the maximum likelihood estimates of the parameters. When the stopping tolerance on the expected improvement is satisfied, the maximum of the weighted average of several expected improvement functions based on different estimates of the parameters is computed, and if this exceeds the stopping tolerance a sample point is added where the maximum of the weighted average expected improvement was found. Then we continue as before using maximum likelihood parameters. If the maximum of the weighted average expected improvement does not exceed the tolerance, the algorithm stops. The steps of the algorithm are:

1. evaluate the objective function at a set of n initial points $\{\boldsymbol{x}^{(i)}\}_{i=1}^{n}$,

2. maximize the expected improvement function to find a promising new sample point $\boldsymbol{x}^{(n+1)}$. If the maximum expected improvement at this point is greater than the tolerance go to 4,

3. maximize the weighted average expected improvement to find a promising new sample point $\boldsymbol{x}^{(n+1)}$. If the maximum weighted average expected improvement at this point is greater than the tolerance go to 4, otherwise stop

4. evaluate the objective function at the point $\boldsymbol{x}^{(n+1)}$ at which the sampling criterion is maximized,

5. set $n = n + 1$, go to 2.

The weights for the expected improvement parameters were tuned empirically using data from runs of the Txxx problems with 4 and 6 starting points. The aim was to get the best balance between improving the final error and increasing the total number of function evaluations. The result for the 500 test functions

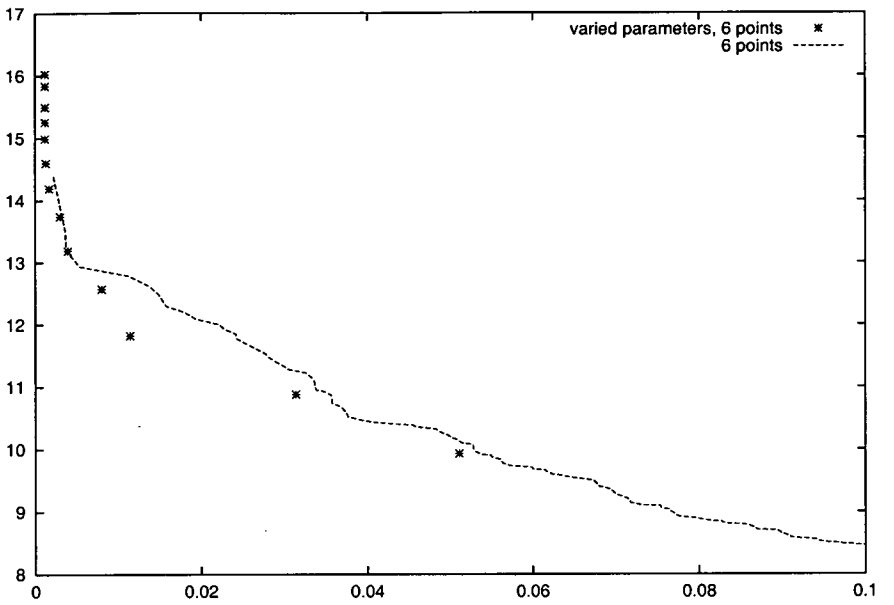

Figure 5.20: Expected improvement with varied parameters for 6 initial points on the test set Txxx

using these tuned weights is shown in Figure 5.20. Compared with the sizes of the problem sets used, i.e. 1000 for tuning and 500 for running, the number of parameters, i.e. 8, is small. Thus overfitting of the parameters is unlikely. The run for 6 starting points using the tuned parameter settings and weighted average expected improvement competes well over the entire range of final average improvements with the normal run for 6 starting points.

An alternative to using a fixed set of weights for the weighted average expected improvement is to calculate the weights using a Bayesian update. We first illustrate this for a situation where there are only two possible values for θ, and where σ and μ are fixed. The set of test functions is 1000 functions, the 500 in T025 and the 500 in T225. The expected improvement used as a sampling and stopping criterion is a weighted average of the expected improvement obtained by using $\theta = 25$ and $\theta = 225$. The weights are obtained as follows: the prior probability that θ is 25 or 225 is 0.5. This is multiplied by the respective likelihood value $L_{25} = L(\theta = 25)$ or $L_{225} = L(\theta = 225)$ and normalized,

$$\begin{aligned} w_{25} &= \frac{0.5L_{25}}{0.5L_{25} + 0.5L_{25}} \\ w_{225} &= \frac{0.5L_{225}}{0.5L_{25} + 0.5L_{25}}. \end{aligned}$$

The weights w_{25} and w_{225} are the posterior probability of the sample having parameters $\theta = 25$ or $\theta = 225$ respectively. Figure 5.21 shows the performance of

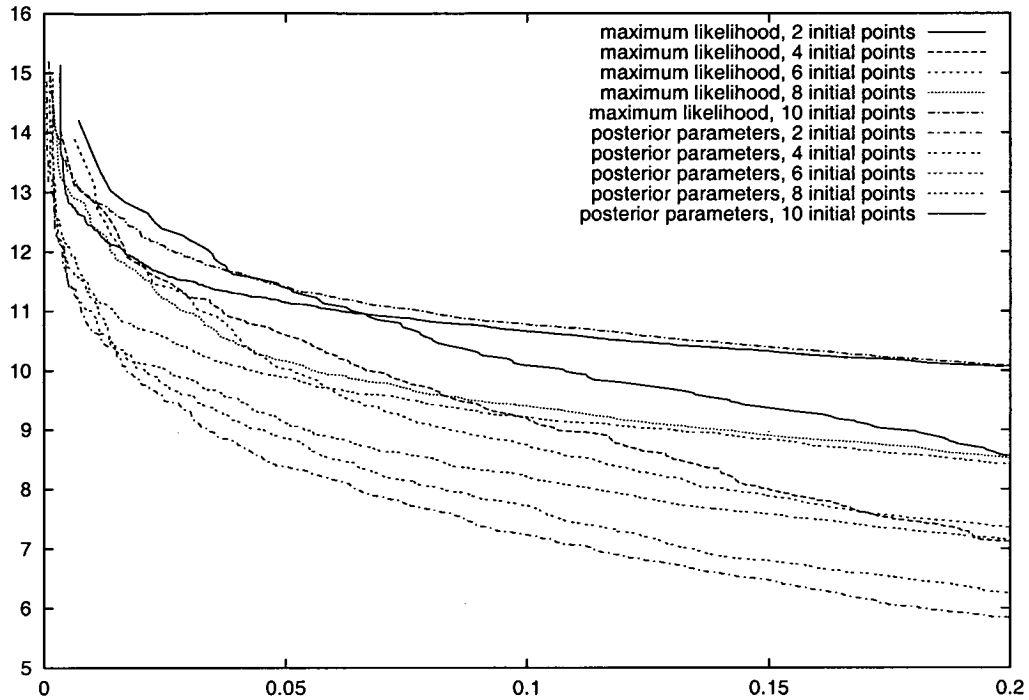

Figure 5.21: Expected improvement variants for different numbers of initial sample points, T025/T225 test functions

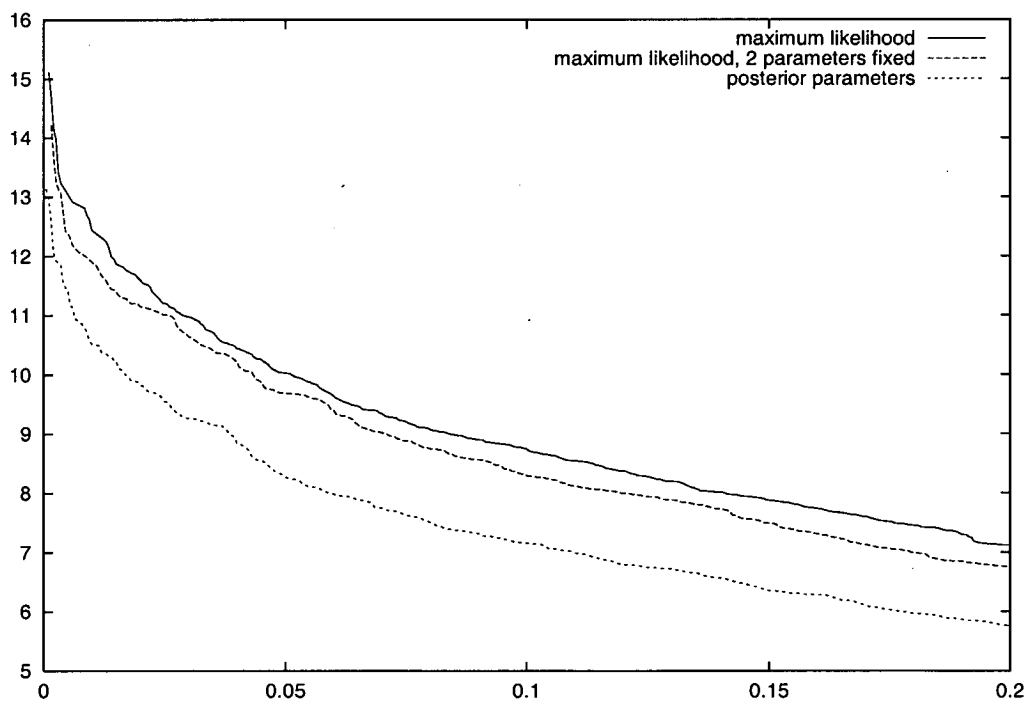

Figure 5.22: Lower envelopes for expected improvement variants for different numbers of initial sample points, T025/T225 test functions

the maximum likelihood variant and the weighted parameter variant for different numbers of starting points. It is clear that the weighted variant for two initial points performs well compared with other runs. To make the difference between the two variants more clear, Figure 5.22 shows the lower envelopes of the curves for maximum likelihood parameters and weighted parameters as in Figure 5.21. Also shown is the curve for a variant with μ and σ fixed to $\mu = 0$ and $\sigma = 1$, and θ estimated by maximum likelihood estimation. This clearly performs better than the maximum likelihood variant estimating μ, σ, and θ, but throughout the method using the weighted parameters achieves the same average final error with fewer function evaluations.

The method of selecting parameter weights by tuning seems quite ad-hoc and although it did produce an improved performance this was not large. On the other hand, selecting parameter weights using a posterior distribution on the θ has a clear theoretical justification and in this example did produce good results. This motivates us to investigate how to calculate the expected improvement using more general posterior distributions for the parameters. The method uses conditional functions, and how to do this is described in the following section.

5.7.2 Expected Improvement by Conditional Functions

Given a set of data points $\{(\boldsymbol{x}^{(1)}, y_1), \ldots (\boldsymbol{x}^{(n)}, y_n)\}$ and a distribution of values for the parameters $\boldsymbol{\theta}$, μ, and σ, a family of conditional functions which interpolate the set of data points can be generated. For each of the functions in this family a curvature bound can be derived and then used to bound the function values on subintervals. Areas where it is clear that the function does not take values below a fixed target value, i.e. the best function value at the sample points, are excluded from further search. This method can be used to find the actual improvement at every point $\boldsymbol{x}$ and also the improvement at the minimum of the conditional function. By averaging over the entire family we can find the average improvement at every point $\boldsymbol{x}$ and the average total improvement, i.e. the average of the improvements at the minima of the conditional functions. If enough such conditional functions are generated these averages should give a good estimate of the expected improvement at every point $\boldsymbol{x}$ and the total expected improvement over all points.

This technique has been used with the Txxx problems described in Chapter 4 4.5. The parameters $\boldsymbol{\theta}$, μ, and σ are sampled using their posterior distributions as in 4.4, and the conditional functions interpolating the given data are generated by the method described in 4.3. The global minimum of each function is found and the maximum improvement and the improvement at each $\boldsymbol{x}$ calculated. These

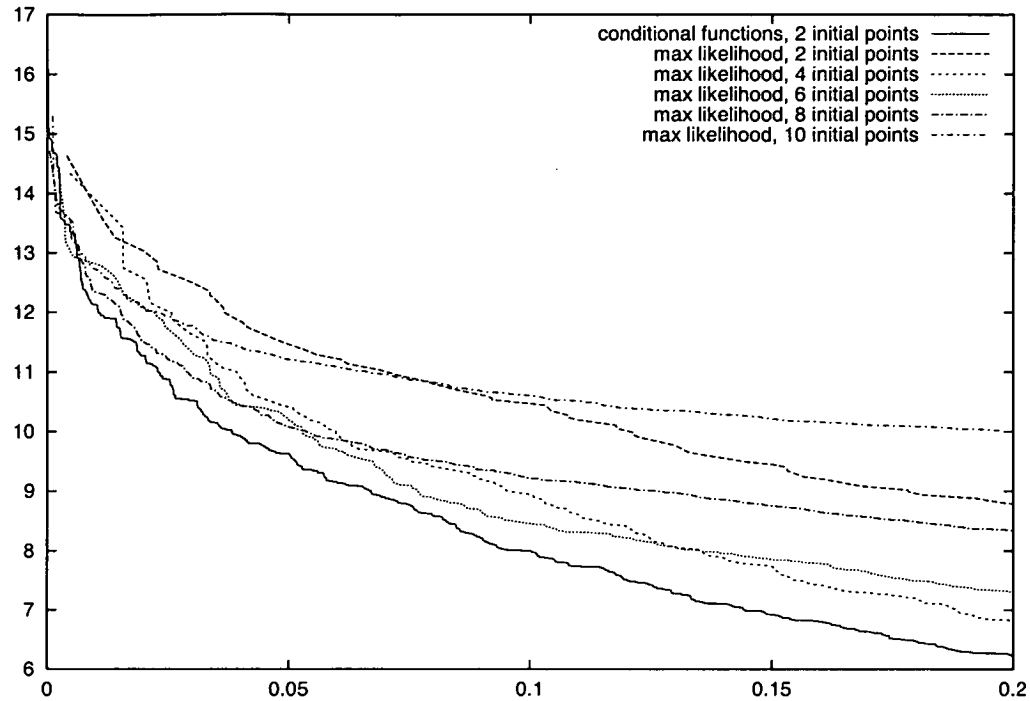

Figure 5.23: Expected improvement using conditional functions versus expected improvement using maximum likelihood, Txxx test functions

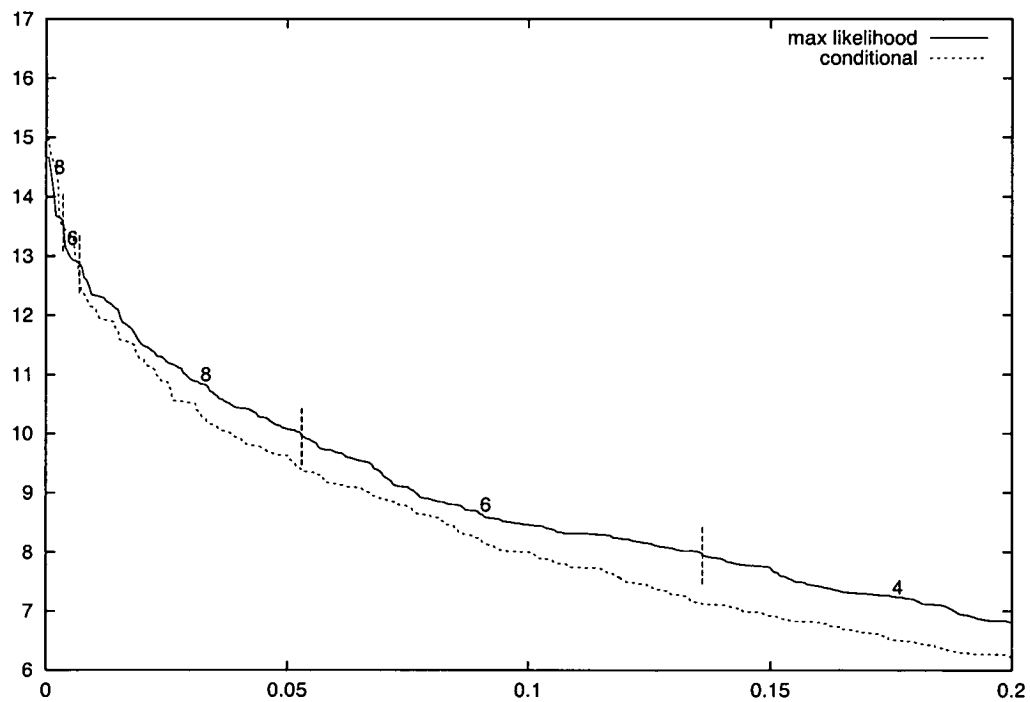

Figure 5.24: Lower envelope of maximum likelihood expected improvement plots, conditional function plot for 2 initial points, Txxx test functions

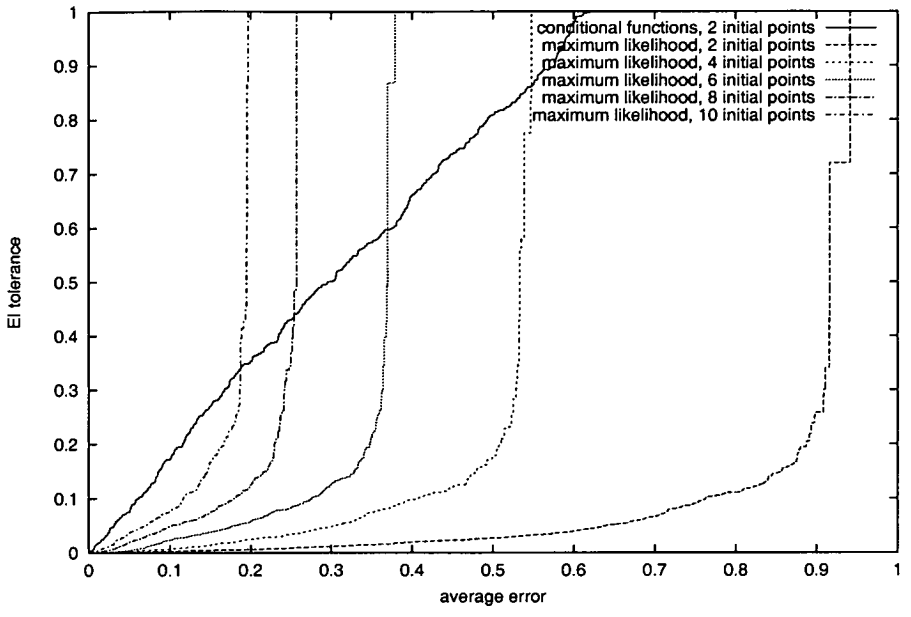

Figure 5.25: Expected improvement stopping tolerance and average final error, Txxx test functions

results are averaged over all the conditional functions to get estimates of the pointwise and total expected improvement.

Like the expected improvement obtained from the maximum likelihood estimates of the parameters, either of these expected improvements based on conditional functions can be used as the stopping criterion, and the pointwise expected improvement can be used for selecting the next sample point. Figure 5.23 compares the normal maximum likelihood variant for different numbers of starting points with a variant using 2 initial points that uses total expected improvement based on conditional functions for stopping and the pointwise expected improvement based on conditional functions for selecting the next sample point. The plot shows the average number of function evaluations plotted against the average final improvement/error on the x-axis, obtained by setting different stopping tolerances. The variant using conditional functions, in terms of efficiency, competes well with the normal runs for different starting points. Most average final errors are achieved with fewer sample points, which is crucial if function evaluations are expensive. Figure 5.24 shows the lower envelope of the maximum likelihood outcomes in Figure 5.23, and the conditional functions results. Indicated on the maximum likelihood envelope are the numbers of initial sample points with which the corresponding errors were achieved with the smallest average total number of sample points. The fact that the expected improvement method based on conditional functions works well for two initial points only is

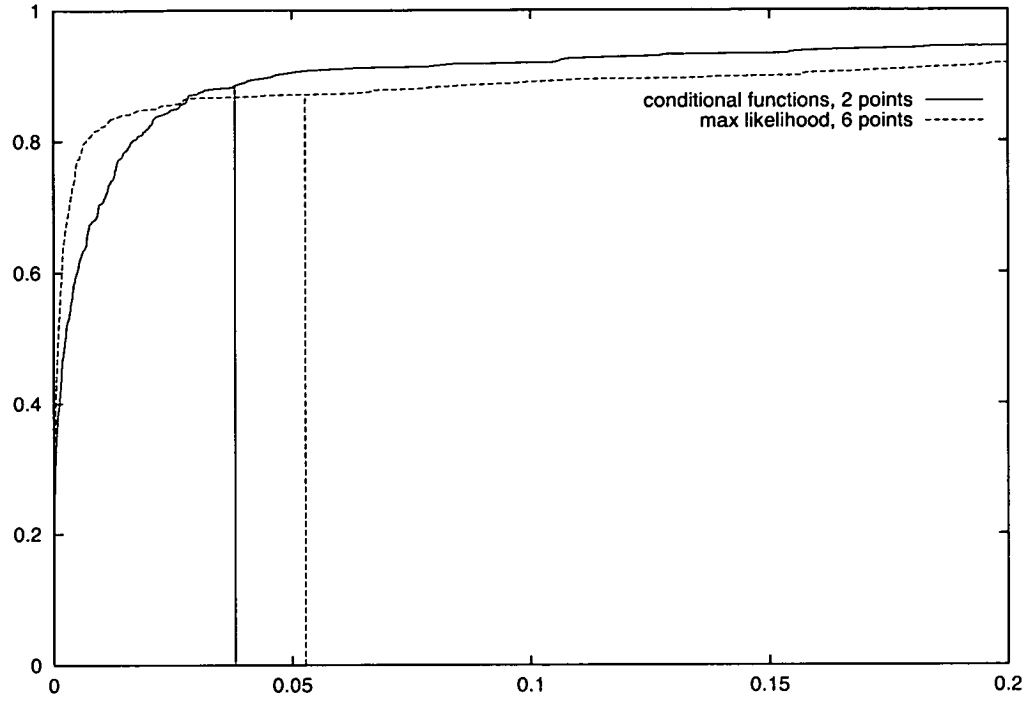

Figure 5.26: Cumulative proportion of the Txxx test functions with an average error the given x-value for a tolerance leading to an average total of ten iterations

encouraging. There is no dilemma here of how many initial points to choose, a very small number suffices to achieve good results. Figure 5.25 shows how the stopping tolerance on the expected improvement and the average final error are related. Clearly, the expected improvement from the conditional functions is a much more reliable predictor of the average improvement possible on termination.

There are two parameters which have to be selected when using the maximum likelihood stopping method: the number of initial points and the stopping tolerance. With this method there is no simple relation between the number of initial sample points or the stopping tolerance set, and the average error achieved. To achieve a desired average error we need to do the following steps. First use a graph of the form of Figure 5.24 to find the optimal number of initial sample points for the desired average error. Then use the graph corresponding to this number of intial sample points in a figure like Figure 5.25 to select the stopping tolerance to achieve the desired average error. In the conditional function variant of the expected improvement there is a much clearer relationship between the final average error and the stopping tolerance.

Previous plots of average number of function evaluations and average final error showed the averages over all 500 test problems in the test set used. It is interesting to know how the different methods compare in terms of the distribution

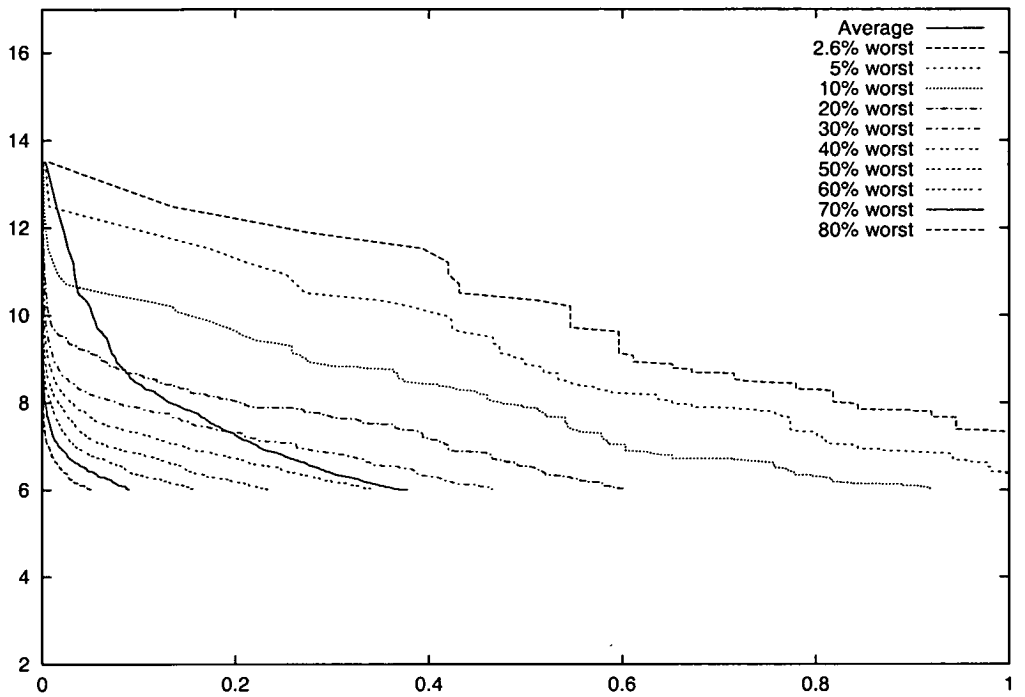

Figure 5.27: Percentage of worst errors for maximum likelihood with six starting points

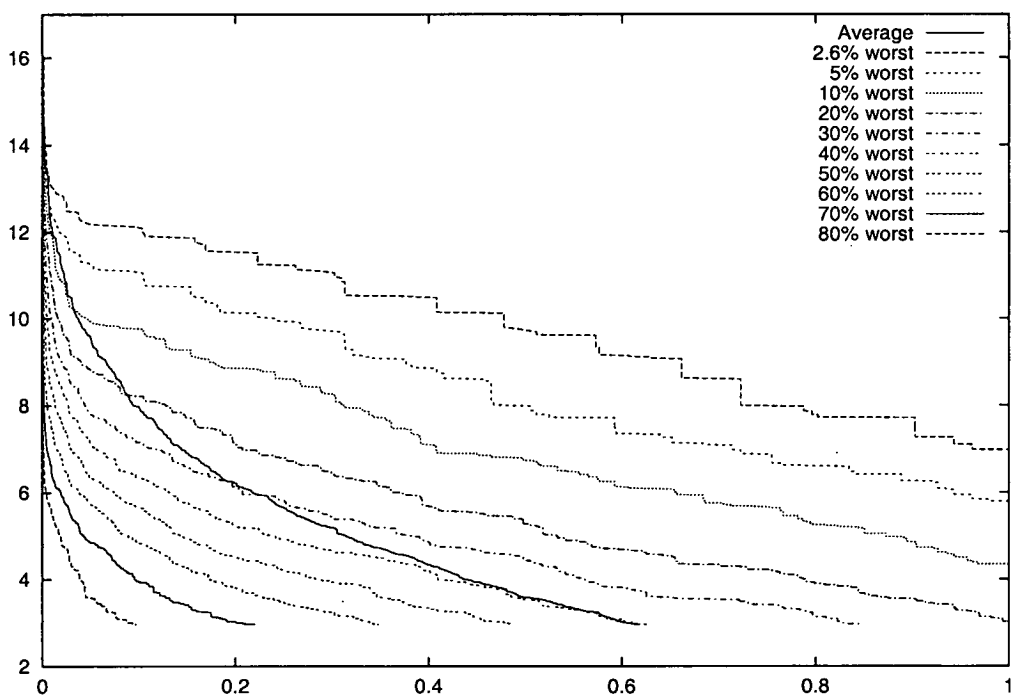

Figure 5.28: Percentage of worst errors for conditional functions with two starting points

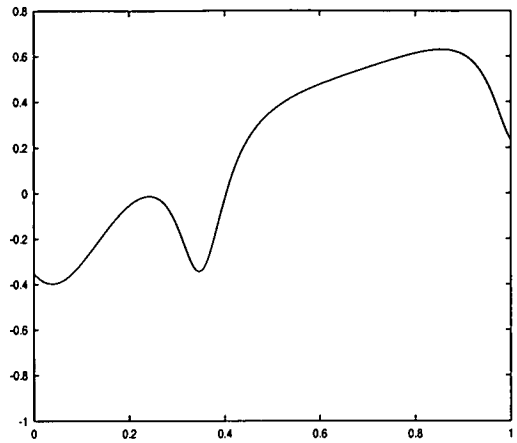

Figure 5.29: Test function 1

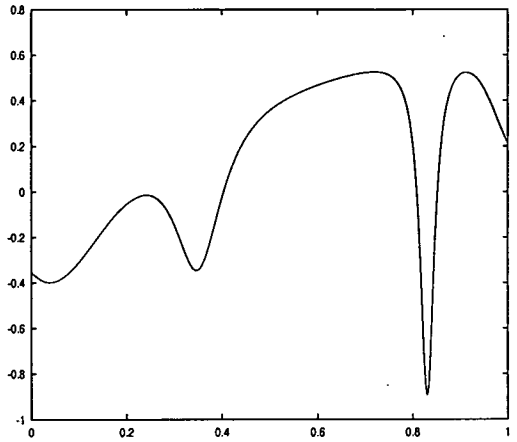

Figure 5.30: Test function 2

of errors about the average, and the number of problems with large errors. For an investigation of this we look at runs using conditional functions with two starting points and runs using maximum likelihood estimation with six starting points — the maximum likelihood runs with six starting points performed well on average.

For a stopping tolerance for each method that achieves an average number of ten iterations, Figure 5.26 shows the cumulative proportion of errors for the runs using conditional functions and for the runs using maximum likelihood estimation. Shown are the sizes of errors on the x-axis and the proportion of functions with an error less than this on the y-axis; the vertical line shows the average error. It can be seen that the maximum likelihood runs have more large final errors and more small final errors, therefore a bigger error variance, compared with the conditional functions runs.

Figures 5.27 and 5.28 show the average of the two variants for all 500 Txxx problems. These are the same as the curves shown in Figure 5.23. Superimposed are curves showing the distribution of the errors. The plots are obtained as follows: for a fixed stopping tolerance, the average number of function evaluations is found, and the worst errors which occurred in all 500 problems. The 10% line in the plot indicates the lower bound of the 50 of 500 worst errors that occurred. It can be seen that the maximum likelihood variant with six starting points has more large final errors than the conditional functions variant.

5.7.2.1 Non-Stationary Test Functions

Examples of a different kind of test functions, i.e. not realizations of stationary Gaussian stochastic processes, are shown in Figures 5.29 and 5.30. Compared with test function 1, shown in Figure 5.29, test function 2, shown in Figure 5.30, has an added deep narrow dip and therefore a different global optimum. While

	Test Function 1			Test Function 2		
variant	best pt	no pts	min value	best pt	no pts	min value
conditional 2 pts	7	14	-0.39733771	21	41	-0.89370861
max LLH 2 pts	10	15	-0.39737467	4	6	-0.35310863
max LLH 4 pts	11	13	-0.39737468	11	12	-0.39816149
max LLH 6 pts	10	10	-0.39737467	10	11	-0.39816137
max LLH 8 pts	10	11	-0.39737508	12	12	-0.39816047
max LLH 10 pts	14	17	-0.39737495	15	23	-0.39816137

Table 5.3: Results for test functions 1 and 2

the expected improvement method based on maximum likelihood works well on test function 1, and uses only small numbers of iterations, it misses the dip and therefore the global optimum in test function 2 almost entirely. The expected improvement method based on conditional functions in this case picks up the dip in test function 2, and finds the global optimum. Results of runs on these functions are presented in Table 5.3, shown are the number of the iteration in which the best point was found, the total number of iterations, and the best function value found. For reference, the minimum value of function 1 is approximately -0.3973752124, and the minimum value of function 2 is approximately -0.8937086124.

A function of this form is very unlikely to be generated from a stationary Gaussian process. The curvature around the narrow dip is much higher than anywhere else, hence the appropriate value for θ is much higher locally around the dip, than it is anywhere else. If it happens that no sample point falls deep enough into the dip, the value of θ will have a low estimate, whereas if points are sampled from the dip, the estimate of θ will be much higher. The maximum likelihood cases fail to sample deep enough in the dip and have low estimates of θ, for example the final estimate of θ in the run using maximum likelihood estimation with 8 initial sample points is approximately 56.5. In contrast, the conditional functions method finds a point nearer the bottom of the dip after 19 function evaluations, and the maximum likelihood estimate of θ increases from about 146 to about 3300. This leads to samples being taken fairly uniformly throughout the interval, hence the large number of sample points. Figures 5.31 and 5.32 show the function and the sample points chosen by the two different methods investigated. Figure 5.31 also shows the BLUP as a dashed line.

5.8 Conclusions

One of the issues discussed in the chapter is how to perform the maximization of the expected improvement function as a sampling criterion. It would be reassuring to be able to maximize the expected improvement to guaranteed global optimality,

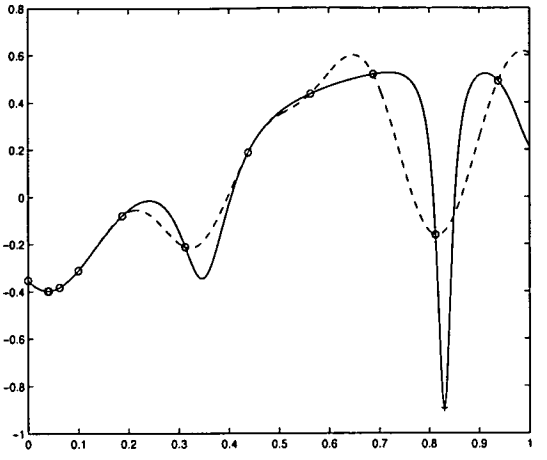

Figure 5.31: Final situation in the optimization of test function 2, maximum likelihood with 8 initial points

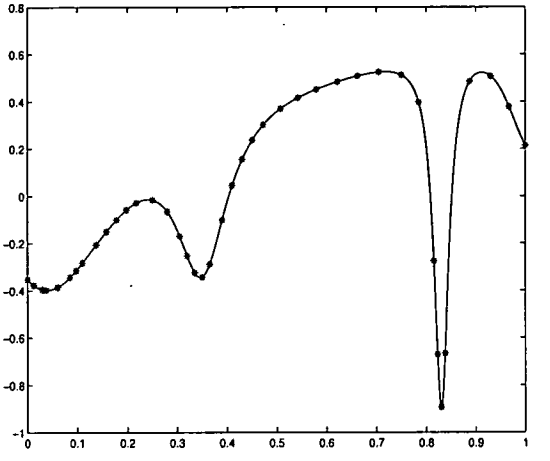

Figure 5.32: Final situation in the optimization of test function 2, conditional functions with 2 initial points

and some improvements in how to do this are pointed out. However, this remains a difficult and time consuming task, and there is little evidence that it is really necessary.

When the parameters are estimated by maximum likelihood it is not straightforward how to choose the best number of starting points or find the stopping tolerance to achieve the desired accuracy. We have described a systematic method for carrying this out, however this is a complex process which would have to be repeated for each new family of functions. The problem with the maximum likelihood estimator is that it is a single point estimate and it is often misleading, and as a consequence the expected improvement often underestimates the actual improvement.

The benefit of expected improvement by conditional functions is that it takes into account the uncertainty in the parameters, whereas normal expected improvement using just one set of parameter estimates, as with maximum likelihood, can be inaccurate. The conditional function approach results in a more predictable algorithm with a better average performance in terms of number of function evaluations. It is also less likely than the normal expected improvement criterion to significantly underestimate the real improvement. It also takes care of the choice of number of initial points in the cases investigated here. Further, there is a more straightforward relation between the expected improvement stopping tolerance and the average final error, which means a stopping tolerance could be chosen to meet a certain required accuracy in the result.

Because of these advantages the conditional function approach is preferable when function evaluations are expensive. However, as implemented here the con-

ditional function approach does require more work per iteration than the maximum likelihood approach, so will not be superior computationally if function evaluations are cheap enough. Further work is needed to find if it can be made efficient enough to be used for more than one dimension.

Unfortunately, when using expected improvement based on maximum likelihood estimates of the parameters, the ranges of optimality for different numbers of initial points and the form of stopping tolerance as a function of average error are very sensitive to the family of functions being minimized. It is therefore necessary to do the analysis separately for each new family. In contrast, using a Bayesian approach to estimating the parameter distributions, is a much simpler process. All that is required is that a prior distribution of the parameters is provided. There is no advantage in this method in using a different number of initial sample points for different desired final average tolerances. Also the curve of expected improvement tolerance as a function of average error has a straight line through the origin, so its gradient is the only parameter needed when selecting the stopping tolerance.

Chapter 6

Generalized Regression and Derivatives

As an extension to the methods that were described in Chapter 3, we investigate here the use of generalized regression in the model of the objective function, and the use of derivatives of the objective function.

6.1 Generalized Regression

In Chapter 3 we introduced the background of the Kriging approach. In particular, in Kriging the objective function is modeled by

$$Y(\boldsymbol{x}) = \sum_{j=1}^{m} \beta_j f_j(\boldsymbol{x}) + \epsilon(\boldsymbol{x})$$

where the first term $\boldsymbol{f}^T\boldsymbol{\beta}$ is a general linear model and the second component $\boldsymbol{\epsilon}(\boldsymbol{x})$ is the departure from the linear model, which is treated as the realization of a stationary Gaussian stochastic process. We also introduced the best linear unbiased predictor (BLUP) and the mean squared error (MSE), and used these to calculate the expected improvement (EI) at any point $\boldsymbol{x} \in [0,1]^d$. As mentioned in Chapter 3, commonly only a trivial regression function $\boldsymbol{f} = 1$, so that $\boldsymbol{f}^T\boldsymbol{\beta} \equiv \mu$, is used. We investigate here how to use general regression functions and compare results for some examples.

Recall from Chapter 3 that the best linear unbiased predictor for a constant regression term is given by

$$\hat{y}(\boldsymbol{x}) = \hat{\mu} + \boldsymbol{r}_x^t R^{-1}(\boldsymbol{y} - \mathbf{1}\hat{\mu})$$

and the mean squared error is

$$s^2(\hat{y}(\boldsymbol{x})) = \sigma^2 \left[1 - \boldsymbol{r}_x^t R^{-1} \boldsymbol{r}_x + \frac{(1 - \mathbf{1}^t R^{-1} \boldsymbol{r}_x)^2}{\mathbf{1}^t R^{-1} \mathbf{1}}\right].$$

We also introduced the BLUP and MSE for the general case, where $\boldsymbol{f}$ contains known regression functions. The BLUP for generalized regression is

$$\hat{y}(\boldsymbol{x}) = \boldsymbol{f}_x^t\hat{\boldsymbol{\beta}} + \boldsymbol{r}_x^t R^{-1}(\boldsymbol{y} - F\hat{\boldsymbol{\beta}})$$

and the MSE is

$$s^2(\hat{y}(\boldsymbol{x})) = \sigma^2\left[1 - (\boldsymbol{f}_x^t, \boldsymbol{r}_x^t)\begin{pmatrix} 0 & F^t \\ F & R \end{pmatrix}^{-1}\begin{pmatrix} \boldsymbol{f}_x \\ \boldsymbol{r}_x \end{pmatrix}\right]$$

and we showed that this can also be written as

$$\begin{aligned} s^2(\hat{y}(\boldsymbol{x})) &= \sigma^2[1 - \boldsymbol{r}_x^t R^{-1}\boldsymbol{r}_x + \boldsymbol{f}_x^t(F^tRF)^{-1}\boldsymbol{f}_x - 2\boldsymbol{r}_x^t R^{-1}F(F^tRF)^{-1}\boldsymbol{f}_x \\ &\quad + \boldsymbol{r}_x^t R^{-1}F(F^tRF)^{-1}F^tR^{-1}\boldsymbol{r}_x]. \end{aligned} \tag{6.1}$$

For the constant regression case we use the gradients and Hessian matrices of the BLUP and MSE to evaluate the BLUP and MSE at any point $\boldsymbol{x} \in [0,1]^d$ by $c + \boldsymbol{g}^t\boldsymbol{x} + \frac{1}{2}\boldsymbol{x}^tH\boldsymbol{x}$, where c is the constant term, $\boldsymbol{g}$ the gradient, and H the Hessian matrix of the respective function. In the case when generalized regression is used the linear algebra gets more difficult, and we know from our calculations in Chapter 5 that the Hessian matrix of the MSE function is not generally negative semi-definite. This in particular makes using branch and bound to find the maximum EI by finding upper bounds on the MSE s^2 and lower bounds on the BLUP $\hat{y}$ more difficult, as it requires maximizing the MSE function s^2. So in the case of generalized regression we evaluate the BLUP $\hat{y}$ and the MSE s^2 by other means as shown below, and do not use branch and bound for the maximization of the expected improvement, but only a grid search, and a local search from the best point found in the grid search. In particular this means that the MSE function does not have to be maximized to find an upper bound for the expected improvement. The BLUP, in terms of $\boldsymbol{f}_x$ and $\boldsymbol{r}_x$, which both depend on $\boldsymbol{x}$, is of the form

$$\hat{y}(\boldsymbol{x}) = \boldsymbol{f}_x^t\boldsymbol{\beta} + \boldsymbol{r}_x^t\boldsymbol{\rho}$$

and we can find the m-vector $\boldsymbol{\beta}$ and the n-vector $\boldsymbol{\rho}$ by solving the following augmented system

$$\begin{pmatrix} 0 & F^t \\ F & R \end{pmatrix}\begin{pmatrix} \boldsymbol{\beta} \\ \boldsymbol{\rho} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \boldsymbol{y} \end{pmatrix}. \tag{6.2}$$

The second set of n equations in (6.2)

$$R\boldsymbol{\rho} + F\boldsymbol{\beta} = \boldsymbol{y}$$

is the matrix form of the n equations

$$\hat{y}(\boldsymbol{x}^{(i)}) = \boldsymbol{f}_{x^{(i)}}^t \boldsymbol{\beta} + \boldsymbol{r}_{x^{(i)}}^t \boldsymbol{\rho} = y_i, \quad i = 1, \ldots, n$$

which state the condition that $\hat{y}$ interpolates the objective function in every previous sample point. The next set of m equations corresponds to the unbiasedness constraint. The system (6.2) is not solvable if $m > n$. This is obvious from the structure of the augmented matrix, which is singular if $m > n$.

To find the MSE for general regression we can use expression (6.1) or solve an augmented system similar to the one in (6.2), derived from the expression (6.1) for the MSE. This augmented system is of the form

$$\begin{pmatrix} 0 & F^t \\ F & R \end{pmatrix} \begin{pmatrix} \boldsymbol{q}_x \\ \boldsymbol{t}_x \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}_x \\ \boldsymbol{r}_x \end{pmatrix} \tag{6.3}$$

and the MSE is given by

$$s^2(\hat{y}(\boldsymbol{x})) = \sigma^2 \left[1 - \begin{pmatrix} \boldsymbol{f}_x^t & \boldsymbol{r}_x^t \end{pmatrix} \begin{pmatrix} \boldsymbol{q}_x \\ \boldsymbol{t}_x \end{pmatrix} \right].$$

To solve the two augmented systems (6.2) and (6.3) we use NAG routines [52]: the matrix of the augmented system is factorized into LU factors and these are then used to solve the systems.

What we want to investigate here is whether for an objective function with a clear trend, the algorithm works better with regression functions capturing this trend. What is also of interest is how such an algorithm performs on an objective function with no trend.

As a particular example for the regression functions in $\boldsymbol{f}$ we will in the following consider a quadratic polynomial basis, such that in one dimension $\boldsymbol{f} = (1, x, x^2)^t$ and in two dimensions $\boldsymbol{f} = (1, x_1, x_2, x_1x_2, x_1^2, x_2^2)^t$.

6.1.1 Computational Results

The test functions used for the regression experiments are the Txxx sample paths of Gaussian stochastic processes as described in Section 4.5 of Chapter 4, with quadratic terms added. In one dimension the added quadratic terms are of the form $\lambda(x-0.5)^2 = \lambda x^2 - \lambda x + \frac{\lambda}{4}$ and vary in their curvature 2λ. The quadratic regression term in the stochastic process model is of the form $\boldsymbol{f}_x^t\boldsymbol{\beta} = a_0 + a_1 x + a_2 x^2$. Table 6.1 shows that, when using quadratic regression for objective functions with a quadratic trend and $\lambda = 25$, i.e. a curvature of $2\lambda = 50$ (problems C33 in the Table), the coefficients a_0, a_1, a_2 are estimated quite accurately, i.e. $a_0 \approx \frac{\lambda}{4}$, $a_1 \approx -\lambda$, and $a_2 \approx \lambda$, so in these cases the quadratic regression term appears to capture the trend in the function well.

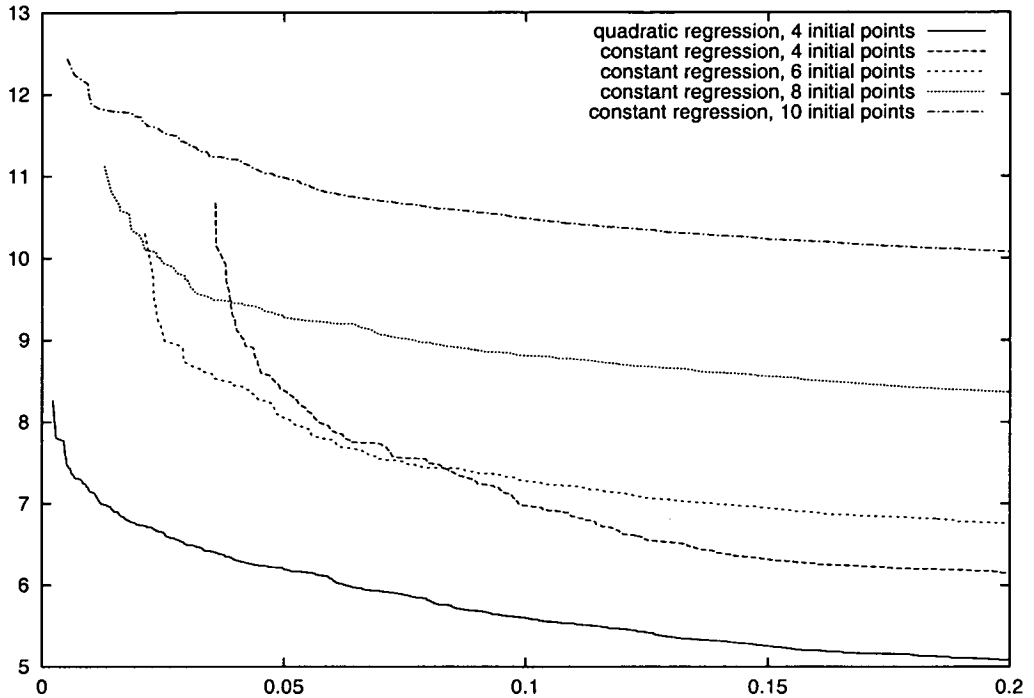

Figure 6.1: Quadratic regression versus constant regression for 500 test functions in Txxx with strong quadratic trend

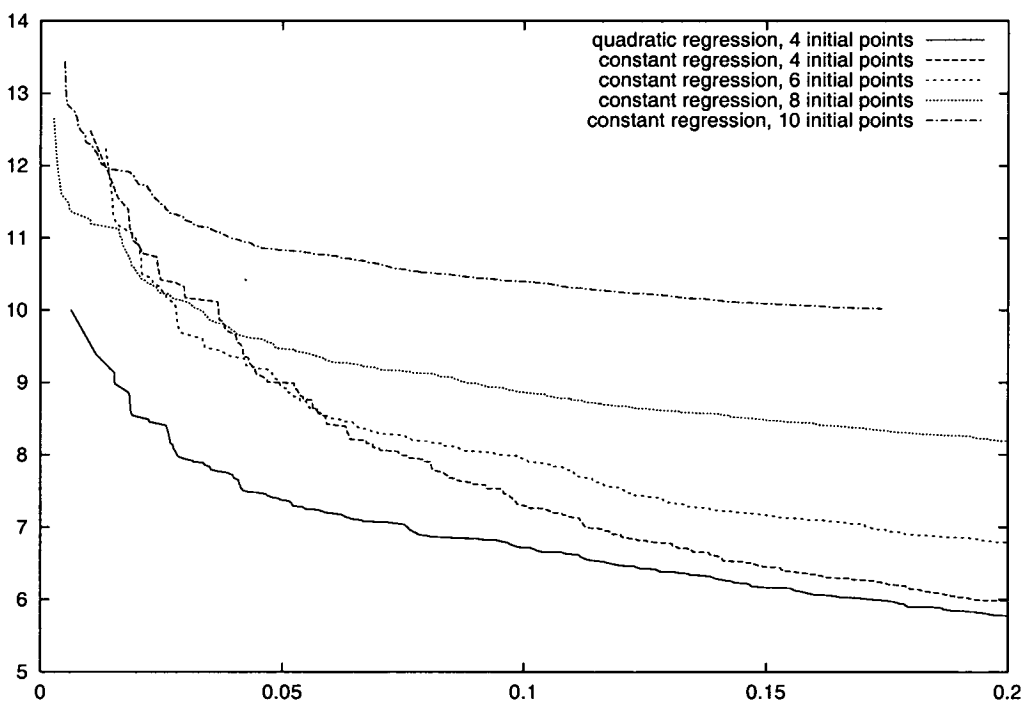

Figure 6.2: Quadratic regression versus constant regression for 500 test functions in Txxx with mild quadratic trend

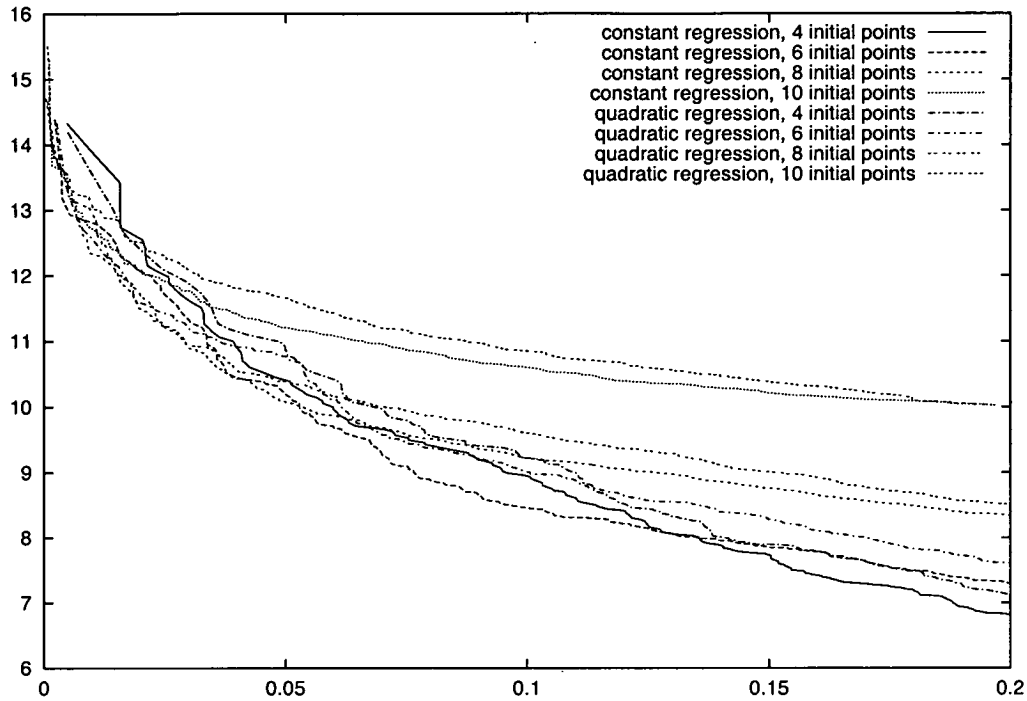

Figure 6.3: Quadratic regression versus constant regression for 500 test functions in Txxx with no trend

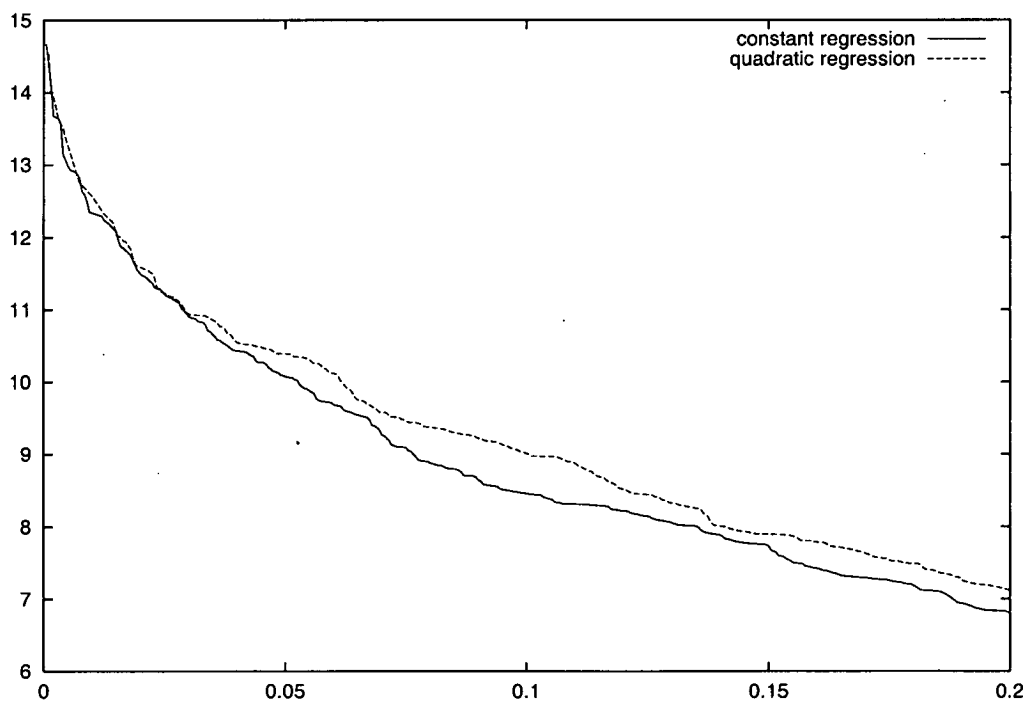

Figure 6.4: Lower envelopes of the plots for the constant and quadratic regression variants for 500 test functions in Txxx with no trend

variant	avgpts	fundist	fin EI	a_0	a_1	a_2
C00/AV0.04	11.976	0.254E-01	0.357E-03	0.736E-01		
C00/AV2.04	11.252	0.360E-01	0.371E-03	0.398	-1.43	1.37
C33/AV0.04	10.894	0.194E-01	0.210E-03	3.03		
C33/AV2.04	8.568	0.188E-01	0.204E-03	6.45	-25.5	25.5
C00/AV0.06	11.988	0.221E-01	0.359E-03	0.618E-01		
C00/AV2.06	11.944	0.170E-01	0.310E-03	0.228	-0.753	0.735
C33/AV0.06	10.550	0.209E-01	0.252E-03	2.79		
C33/AV2.06	9.796	0.138E-01	0.184E-03	6.29	-24.8	24.8

Table 6.1: Comparison of variants using constant and using quadratic regression for test functions with no trend and test functions with quadratic trend

For a quadratic term with a large curvature of 200, added to the 500 test functions in Txxx, some results are shown in Figure 6.1. Plotted are the average total number of points against the average improvement/error. The best results are clearly obtained when using 4 initial points and quadratic regression, such that $\boldsymbol{f} = (1, x, x^2)$. This requires fewer iterations and achieves better results than the variant with constant regression for 4, 6, 8, or 10 initial points. For a smaller curvature of 50, the differences between the curves are much less pronounced, but using 4 initial sample points and quadratic regression functions still clearly outperforms the other variants. This is shown in Figure 6.2. For test functions with no trend, Figures 6.3 and 6.4 illustrate that using only a constant regression term performs slightly better than using quadratic regression functions. Figure 6.3 shows the outcome of the constant and the quadratic regression runs for 4, 6, 8, and 10 initial points. It is not very clear here which variant is the most effective, therefore the lower envelopes of the plots for the constant regression case and the quadratic regression case are plotted in Figure 6.4. These lower envelopes show the most efficient variant of the relevant algorithm on different numbers of initial points. Larger average final errors are achieved with fewer sample points in the best cases when using constant regression rather than quadratic, but for smaller average final errors the best cases of the two variants are almost equally efficient.

Table 6.2 shows average results for 12 two-dimensional test functions with no trend (C00) and with quadratic trend (C23). Compared are the variant of the algorithm using a constant regression term (AV0.20) and the variant using quadratic regression (AV2.20). It should be noted that 12 such test functions are a small sample and not representative of all possible such sample paths. In terms of final error the variant with constant regression outperforms the quadratic regression variant on the test functions with no trend, but there is a tendency that the quadratic regression variant takes fewer sample points even here. For

variant	avgpts	$1/\sqrt{\theta}$	$1/\sqrt{\theta}$	fundist	fin EI
C00/AV0.20	47.833	0.131	0.131	0.265E-01	0.764E-03
C00/AV2.20	44.750	0.117	0.122	0.570E-01	0.663E-03
C23/AV0.20	39.583	0.202	0.181	0.376E-01	0.767E-03
C23/AV2.20	28.417	0.120	0.108	0.383E-02	0.349E-03

Table 6.2: Constant regression versus quadratic regression for 12 test functions in two dimensions with no trend, and with quadratic trend

the test functions with quadratic trend the quadratic regression variant achieves a smaller final error with fewer sample points than the constant regression variant. An observation that can be made when running these problems, is that if the quadratic trend in the objective function dominates, then the problems in the constant regression variant tend to become near-singular because of small $\boldsymbol{\theta}$ values. In the variants with regression the regression part of the model picks up the strong trend and the problem does not become near-singular so easily.

6.1.2 Conclusions

In the cases investigated here, the algorithmic variant using constant regression slightly outperforms the variant with quadratic regression for objective functions with no clear trend. If there is a clear quadratic trend in the objective function, the variant using quadratic regression picks this up well and performs considerably better than the variant with constant regression terms. This leads to the conclusion that, as opposed to the apparently existing opinion that generalized regression has no benefits in the Kriging approach, generalized regression may have benefits and may lead to an improvement in efficiency and performance.

6.2 Derivatives

For some objective functions gradients might be readily available, or at least not very expensive to compute by direct evaluation or automatic differentiation. The issue addressed here is whether it is worth the extra effort to incorporate gradient information into the Kriging model. If obtaining gradients involves extra evaluations of the objective functions, which would be the case for example if gradients had to be calculated by numerical methods, then this might be a prohibitively expensive task. On the other hand, in some physical applications for example, gradient information is available. We face a trade-off between the added benefit of using gradients and the cost of obtaining them. Some cases where gradient information is used are investigated, and compared to cases where no gradient

id	error bounds
1	$0.001 < \lvert y_{\min} - y_{\text{gop}} \rvert$
2	$0.005 < \lvert y_{\min} - y_{\text{gop}} \rvert$
3	$0.010 < \lvert y_{\min} - y_{\text{gop}} \rvert$
4	$0.050 < \lvert y_{\min} - y_{\text{gop}} \rvert$
5	$0.100 < \lvert y_{\min} - y_{\text{gop}} \rvert$
6	$0.500 < \lvert y_{\min} - y_{\text{gop}} \rvert$

Table 6.3: Number codes for failure cases

information is used. The theoretical background of how gradients can be used to achieve a better approximation to the objective function was covered in Chapter 3. Recall that we can use the BLUP and MSE in much the same way as before, only now the BLUP interpolates derivatives as well as function values, and the correlation matrix R contains as entries also the relevant derivatives of the correlation function.

When comparing different runs of the algorithm we sometimes use failure codes as given in Table 6.3 for the difference $|y_{\min} - y_{\text{gop}}|$ of the best objective function value found $y_{\min}$, compared to the actual global optimum of the objective function y_{gop}.

6.2.1 Computational Results

Table 6.4 shows results of 500 normal runs with different numbers of starting points, and of 500 runs using derivatives with the same number of starting points, all with a stopping tolerance of 0.001 on the expected improvement. As test functions the set of 500 functions Txxx as described in Section 4.5 is used. Displayed are occurrences of errors of type $1 - 6$, as detailed in Table 6.3, average total number of sample points (avgpts), i.e. average number of objective function and derivative evaluations, average difference of the best function value found and the global minimum (fundist) and its standard deviation (distdev), and the average final expected improvement (fin EI). It is clear from the table that the average number of sample points used when optimizing using derivative information is smaller than when not, a so-called "normal run". For example the average total number of sample points in the normal run starting with two initial points (AV0.02) is 12.180, compared with 9.536 in the row underneath for the variant using derivatives starting with two points (AV1.02). If derivatives of the objective function are not directly available, and if finding a derivative value involves evaluating the objective function itself, then this reduction in the average total number of points does not seem to make it worthwhile to use derivatives. If derivatives

variant	1	2	3	4	5	6	avgpts	fundist	distdev	fin EI
AV0.02	108	49	39	30	26	11	12.180	0.0349	0.1971	0.326E-03
AV1.02	65	16	9	6	5	1	9.536	0.0040	0.0360	0.219E-03
AV0.04	93	44	33	23	20	5	11.978	0.0237	0.1608	0.356E-03
AV1.04	55	19	13	4	4	0	9.104	0.0033	0.0299	0.222E-03
AV0.06	100	40	33	24	23	8	11.988	0.0221	0.1302	0.359E-03
AV1.06	63	15	9	3	2	0	9.476	0.0022	0.0229	0.215E-03
AV0.08	84	31	21	12	10	2	12.536	0.0089	0.0732	0.309E-03
AV1.08	59	10	4	2	2	0	10.430	0.0011	0.0095	0.147E-03
AV0.10	92	29	21	7	4	1	13.548	0.0040	0.0320	0.268E-03
AV1.10	62	16	9	1	0	0	12.088	0.0010	0.0043	0.959E-03

Table 6.4: Normal runs compared with runs using derivatives for 2, 4, 6, 8, 10 initial sample points, Txxx test problems

are expensive to compute, the number of derivative evaluations necessary might be too high. On the other hand, the number of failure cases of the algorithm is clearly reduced when derivatives are used. For example, there are 11 cases of missing the actual global minimum of the objective function by more than 0.5 in the 500 normal runs, compared with only one such error in the 500 runs with derivatives. The number of errors in category 4 when using derivatives are one fifth or less of the corresponding number of failure cases in the runs without derivatives. The average amount by which the global optimum is missed is clearly reduced when using derivatives, and the corresponding standard deviations are noticeably smaller, so on average the derivative variant pins down the final error much better for this stopping tolerance. Figure 6.5 shows the average total number of points plotted against the average difference between the best objective function value found and the global optimum for 500 test functions. Again the plot suggests that 10 starting points are too many for these test problems. Most successful on average seem to be the runs starting with 4 or with 6 initial sample points. From the plots it is more clear how the average final error and the average number of sample points are related in each of the run variants. In the case of 4 initial points, for example, it can be seen in Figure 6.5 that a comparison of average number of sample points for a given achieved final error is more in favour of the run using derivatives than the table suggested. An average error of about 0.025, which is achieved with an average of approximately 12 points in the non-derivative run, requires an average total of $7.5 - 8$ points when using derivatives. With an absolute stopping tolerance on the maximum expected improvement of 0.00001, the variants using derivatives achieve smaller final errors on average than the variants without derivatives. However, it is also clear here that the reduction in the number of sample points when using derivatives might be too small to

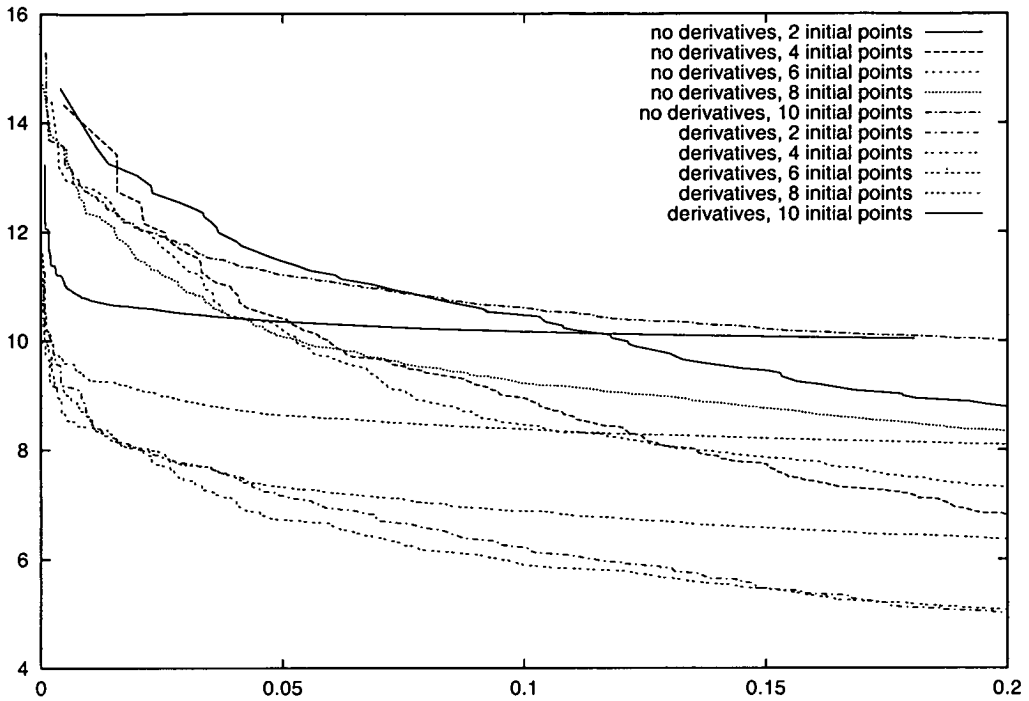

Figure 6.5: Comparison of derivative and non-derivative variant on Txxx test problems

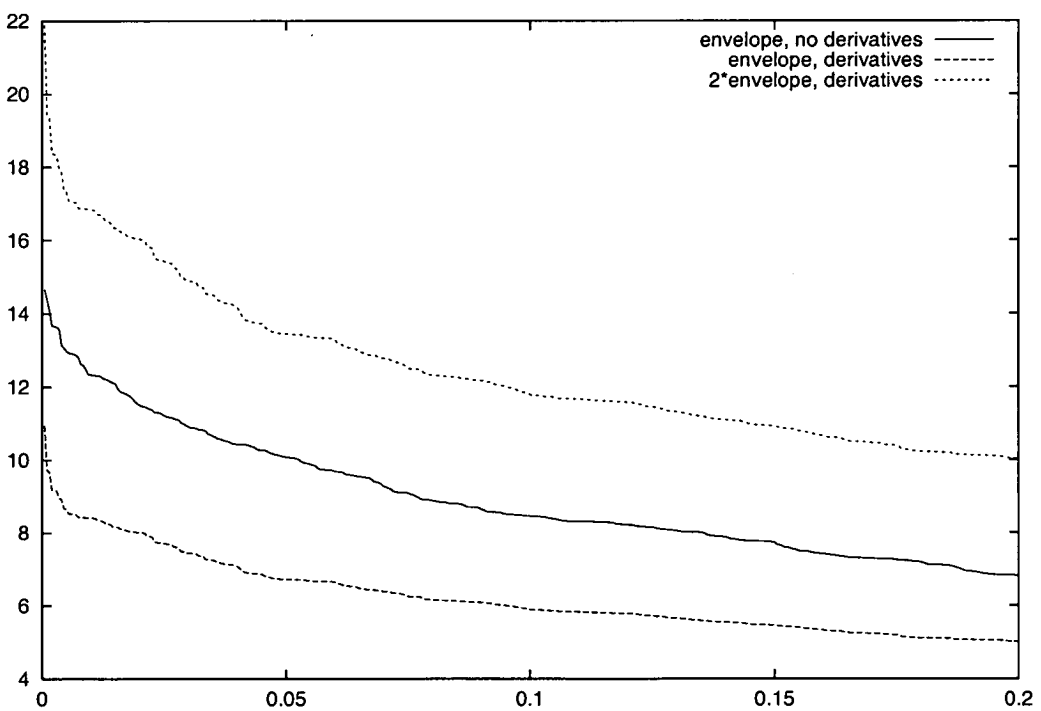

Figure 6.6: Lower envelopes of the plots for derivative and non-derivative variants on Txxx test problems

variant	1	2	3	4	5	6	avgpts	fundist	fin EI	funval
AV0.20	7	2	2	1	1	0	47.833	0.265E-01	0.764E-03	-2.59
AV1.20	7	5	0	0	0	0	34.750	0.320E-02	0.779E-03	-2.62

Table 6.5: Normal variant and variant using derivatives for 12 test functions in two dimensions

make it worthwhile. As before, what we are interested in is how the best cases of the variants compare. This is shown in Figure 6.6, where the lower envelope of the plots for the non-derivative case and the lower envelope of the plots for the derivative case are displayed. Also displayed is two times the lower envelope of the plots for the derivative case. This shows that if derivative evaluations are as costly as function evaluations, and if at every sample point the function as well as the derivative are evaluated, using derivatives is considerably less efficient than using the non-derivative variant of the algorithm. Only if derivative evaluations are less than half the cost of function evaluations might this approach be worth using.

Table 6.5 contains the outcome for 12 test functions in two dimensions for the normal variant and for the variant using derivatives, the test functions are the same as used for Table 6.2. When using derivatives the average final error is clearly reduced, as is the average number of sample points. No failures with code ≥ 3 occur in the runs using derivatives.

Another example in two dimensions is given in Figures 6.7, 6.8, 6.9, 6.10, 6.11. Figure 6.7 shows the contours of an objective function, for which the variant without derivatives fails entirely to find the global minimum, whereas the variant using derivatives successfully finds it. The global minimum of the function is at $(0.6135, 0.2533)$, and is marked by 'x' here. Figure 6.8 shows the contours of the BLUP at the start of the run of the non-derivative variant. This BLUP is based on the initial 20 sample points which are indicated by '$*$'. The global minimum of the objective function again is indicated by 'x'. It becomes clear in Figure 6.9, which shows the stopping situation in this variant, that the global minimum of the function is missed. The sample points which were added in the optimization process are indicated by little squares, and these concentrate around a local minimum, while this algorithmic variant never samples near the global minimum. The contours of the BLUP for the variant using derivatives are shown in Figure 6.10. The starting situation is plotted, with the BLUP based on the indicated 20 initial sample points. The final situation of this variant using derivatives is displayed in Figure 6.11. Indicated are the 20 initial points and the global optimum of the function, as well as the sample points added in the

variant	fc	pts	best	fun dist	pt dist	max EI	value	x_1	x_2
AV0.20	6	40	32	0.920642	0.256165	0.95E-03	-2.600	0.360	0.218
AV1.20	1	30	30	0.003991	0.003585	0.28E-03	-3.516	0.615	0.257

Table 6.6: Example of the outcome of a run in two dimensions, comparing the non-derivative variant and the derivative variant

optimization process; these are again marked by squares. As opposed to the non-derivative variant, the variant using derivatives picks up the global optimum very well, and does so with fewer sample points than the non-derivative variant which fails on this problem. A summary of the outcome of the two runs, not using derivatives and using derivatives, is displayed in Table 6.6. Here fc denotes the failure code. The total number of sample points (pts) and the number of the best sample point (best) are shown, as are the error in function value (fun dist), the distance of the best point from the global minimum (pt dist), the maximum expected improvement (max EI), the best objective function value found, and the best point found. Both variants were run with a stopping tolerance on the expected improvement of 10^{-3}. It becomes clear again, that the non-derivative variant (AV0.20) with a total of 40 sample points fails to find the global minimum of the objective, while the derivative variant (AV1.20) with a total of 30 sample points performs well and achieves a final error of 0.003991, as opposed to an error of 0.920642 in the non-derivative variant.

Note that it can be the case that derivative evaluations are cheaper than function evaluations. A brief example of a function where this is the case is now given.

Example 6.2.1
For an objective function such as

$$y(x) = \sum_{i=1}^{N} \exp(\sin(a_i x)) + \exp(\cos(a_i x))$$

the results of the calculation of the sine, cosine, and exponential from the function evaluation could be stored and reused in the derivative calculation. Therefore the derivative evaluation would be comparatively cheap, and the use of derivatives probably worthwhile.

6.2.2 Conclusions

When using derivatives the number of sample points and therefore the number of objective function evaluations is clearly reduced on average. However, the benefit

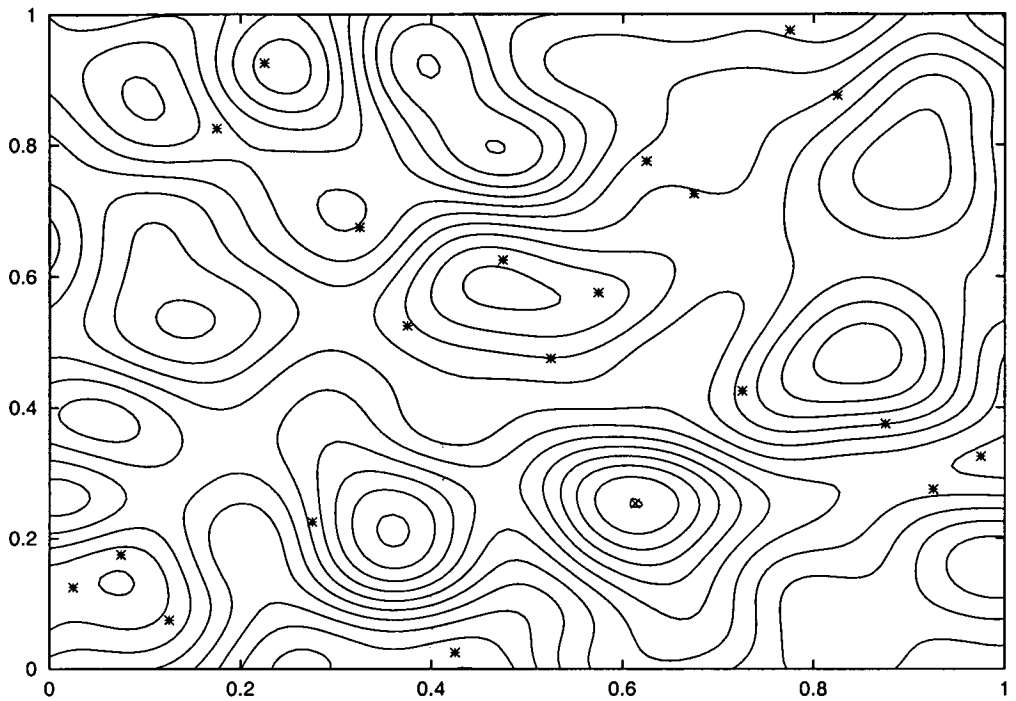

Figure 6.7: Objective function, 20 initial sample points

of using derivatives depends on the cost of obtaining them. If derivatives are calculated at every sample point and if an evaluation of a derivative is as costly as a function evaluation, then using derivatives is probably more expensive than it is worth. Only if derivative evaluations are noticeably cheaper than function evaluations is this approach of incorporating derivatives into the Kriging model worth using. It might be worth trying to evaluate and use derivatives only for certain sample points, thereby improving the model, while not spending too much resources on derivative calculations.

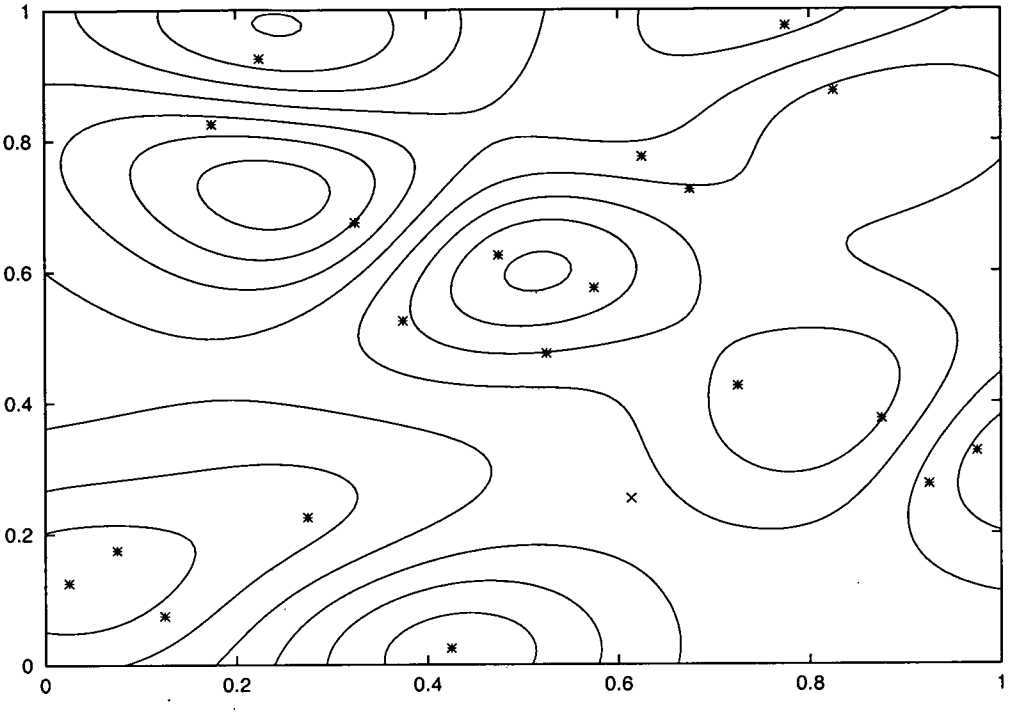

Figure 6.8: BLUP, normal variant, 20 initial sample points

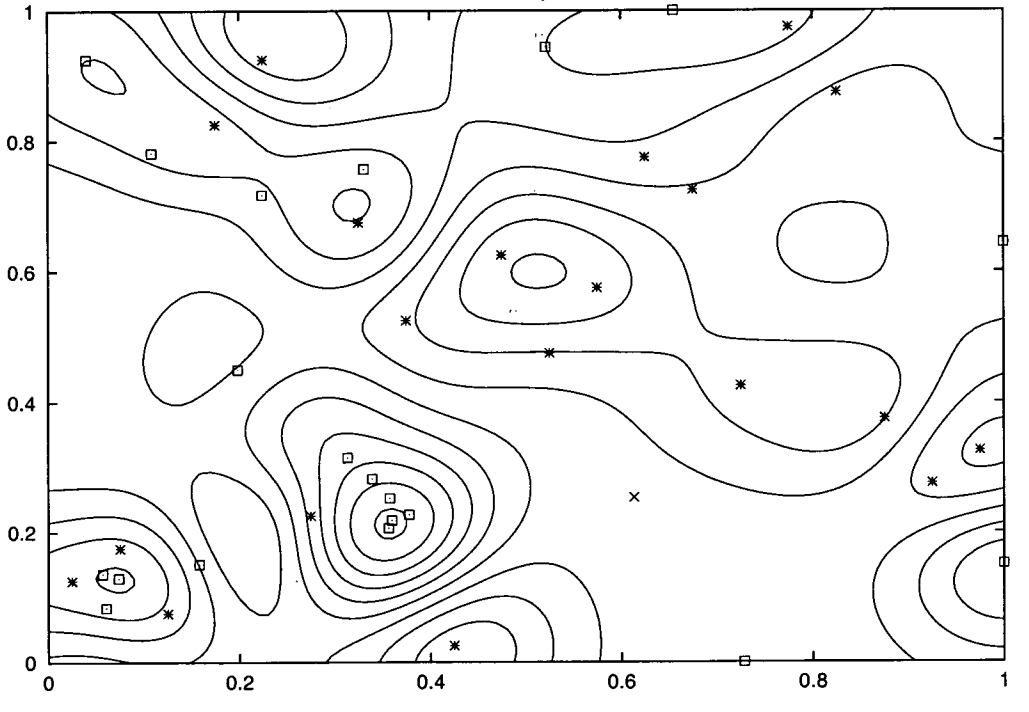

Figure 6.9: BLUP, normal variant, 20 initial sample points, 40 points in total

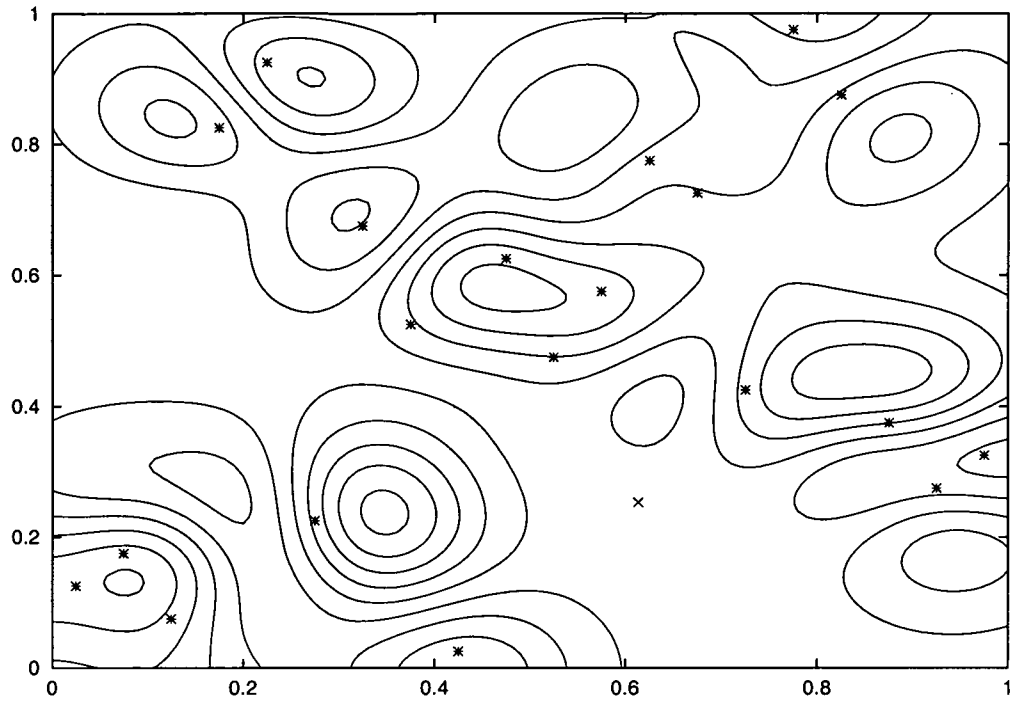

Figure 6.10: BLUP using derivatives, 20 initial sample points

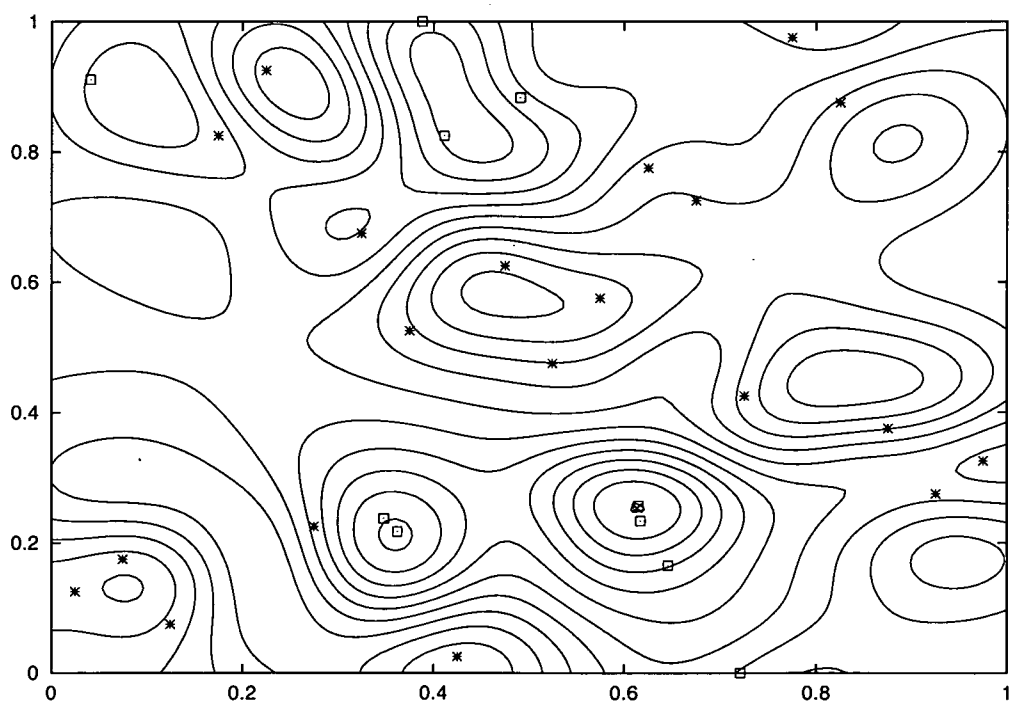

Figure 6.11: BLUP using derivatives, 20 initial sample points, 30 points in total

Chapter 7

Conclusions and Possible Future Work

To round off the work presented in this thesis, we conclude with a summary of achievements, and summarize conclusions drawn from the results of the global optimization method examined. Further, some open questions are identified and ideas for possible future work are given.

7.1 Conclusions

The random function approach to global optimization seems to be well suited to the optimization of expensive functions, and the expected improvement criterion a promising criterion for finding new sample points. The kind of random function optimization method investigated in this thesis requires many auxiliary calculations in every iteration of the optimization process. However, these can be justified when the objective function is expensive to evaluate, since they generally lead to the finding of more efficient sample points.

The Kriging approach assumes that the objective function can be treated as a sample path of a Gaussian stochastic process. Therefore the examined expected improvement method, which uses this approach, should be particularly well suited to such sample paths. For the algorithm these should be the ideal objective functions: if it does not perform well on these, how can it be expected to perform well on other objective functions? This is why such sample paths have been used as test functions here. A way of generating sample paths of stochastic processes has been introduced and explained. Generating these functions can be automated. This makes it possible to generate substantial numbers of functions to test the algorithm on, and to get a feel for some statistics: what sort of functions or sample paths are more likely outcomes of the stochastic process, and how many failure cases of the algorithm occur when applied to a large set of functions. Conditional

functions, i.e. sample paths interpolating the given data, show that it can be possible to turn failure cases of the algorithm into successful runs when taking into account the uncertainty in the estimated parameters.

The expected improvement function is used as a sampling criterion, and as such it might not be necessary to optimize it to global optimality. However sometimes, if the expected improvement is not maximized to global optimality, the stopping criterion is met and the algorithm terminates. Finding the true global maximum and adding another point at it could yield a smaller function value and lead to a better result. If the expected improvement is to be maximized to global optimality, one method of doing this is branch and bound. It has been shown that minimizing the negative mean squared error function (without taking into account any constraints here) is a convex problem and not so difficult to solve as previously thought, which simplifies the bounding of the expected improvement. New ways of improving the bounds for the expected improvement have been developed, and these have been shown to improve the performance of the branch and bound algorithm. Despite these improvements it seems that there is a possibility that branch and bound might still be too expensive as a method of maximizing the expected improvement. A lot of branching is required before useful bounds are obtained and boxes can be discarded. Therefore we have resorted to using a grid and local search for finding a local maximum of the expected improvement, and this gives good results in practice.

Usually, maximum likelihood estimates of the parameters are used in the model of the objective function and to calculate the expected improvement. These estimates often do not reflect the properties of the objective function well, and the expected improvement as a result is underestimated, leading to a premature termination of the algorithm. It has been shown that taking into account the uncertainty in the estimated model parameters leads to good results, and these methods perform better than the standard maximum likelihood method. Two methods which take into account the uncertainty in the estimated model parameters when calculating the expected improvement have been developed and comparative results have been presented. An ad-hoc method of shifting the parameters to calculate a weighted average expected improvement has been used as a stopping rule. In the other method a Bayesian approach is taken and the expected improvement is found by generating conditional functions. Both these approaches take into account that for a smaller number of sample points the estimates of the model parameters are often not accurate, and that there is more uncertainty in the estimates, than when more data points are available. The method based on expected improvement by conditional functions works well for

a small number of starting points, which makes it more reliable in terms of a good choice of number of initial sample points. A recommendation for a sensible number of initial points for the expected improvement criterion based on maximum likelihood estimates of the parameters does not seem to be possible. When using the method based on conditional functions, there is also a clear correlation between the set stopping tolerance and the average final error. This is beneficial since a requested accuracy can be achieved by setting the stopping tolerance to the appropriate value. These results are very useful in practice, since they can solve the issue of how many initial points to use and what stopping tolerance to set.

It has been argued by Jones *et al.* in [34] that the Kriging model is very powerful and that the regression term can be dispensed with. The results in Chapter 6 suggest that this may not be the case. The use of constant and of quadratic polynomial regression functions have been compared for test functions with no trend, and test functions with a quadratic trend. The more dominating the quadratic trend in the objective function, the better the algorithm using quadratic polynomial regression functions performs compared to the algorithm using only a constant regression term. For functions with no trend, using the constant regression term seems to work slightly better.

The benefit of using derivatives in the Kriging model has been investigated. The number of necessary function evaluations is substantially reduced, but derivative evaluations are required. The value of using derivatives therefore depends very much on the cost of obtaining the derivative information.

It has been attempted in the past to make a statement about how many initial sample points should be used for these algorithms. If nothing is known of the structure of the objective function, there is no sensible guideline in terms of overall performance and achieving a good end result of the expected improvement algorithm. As we have shown, knowing something about the structure of the problem can make it easier to solve. Despite the advances developing a general global optimization method to solve any black box function is not an easy task.

7.2 Future Work

There are several open questions and issues which allow for future work to be done. How likely is it that functions which arise from practical applications or functions which are commonly used as test functions could be generated from the stochastic processes being considered? So how well does the stochastic process model used in the optimization method capture and match these functions? How

plausible is this model, are such test or real-life functions describable by a single stationary model, and what is the evidence for or against this? Would it be more appropriate to think about model parameters having local validity, rather than global? So should the model take this into account by for example letting the parameters vary in different regions? In our experiments we have concentrated on the exponential autocorrelation function with smoothness parameters $p_i = 2$. Generating functions using other smoothness parameters, or generating functions from more generally parameterized autocorrelation functions and experimenting with these new functions could be interesting.

Another issue closely related to the validity of the model is the estimation of the parameters. We have pointed out that the estimates of the parameters are often inaccurate. The method investigated so far chooses the next sample point solely to maximize the expected improvement. However, a different choice may be better for producing a narrower posterior distribution for the parameters, so may ultimately lead to a better solution. Is it possible to select sample points to give better estimates of the parameters? If so, this could be incorporated into a merit function for finding new sample points, with the aim of getting a good balance between finding good function values and improving the global validity of the model and the approximating function by improving the estimates. It also might be interesting to further investigate the use of the posterior distributions of the parameters $\boldsymbol{\theta}$, μ, and σ.

There is scope for improvement in the global maximization of the expected improvement function. However, it is not clear how effective a more efficient calculation of the maximum expected improvement would be, and whether maximizing to global optimality brings with it an improvement of the optimization method.

Bibliography

[1] N. M. Alexandrov, R. M. Lewis, C. R. Gumbert, L. L. Green, and P. A. Newman. Optimization with variable-fidelity models applied to wing design. In *Proceedings of the 38th Aerospace Sciences Meeting & Exhibit*, January 2000.

[2] T. W. Anderson. *An Introduction to Multivariate Statistical Analysis.* Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, Inc., 2nd edition, 1984.

[3] I. P. Andraloukis, C. D. Madranas, and C. A. Floudas. αBB: A global optimization method for general constrained nonconvex problems. *Journal of Global Optimization*, 7:337–363, 1995.

[4] M. Björkman and K. Holmström. Global optimization of costly nonconvex functions using radial basis functions. *Optimization and Engineering*, 1:373–397, 2000.

[5] C. G. E. Boender and H. E. Romeijn. Stochastic methods. In R. Horst and P. M. Pardalos, editors, *Handbook of Global Optimization*, volume 2 of *Nonconvex Optimization and Its Applications.* Kluwer Academic Publishers, 1995.

[6] A. Booker, P. D. Frank, G. Shubin, N. Alexandrov, J. Dennis, R. M. Lewis, V. J. Torczon, and M. Trosset. Optimization using approximation models. Center for Research on Parallel Computation newsletter, Winter 1996. http://www.crpc.rice.edu/CRPC/newsletters/-win96/wip.optimization.html.

[7] A. J. Booker, J. E. Dennis, Jr., P. D. Frank, D. B. Serafini, and V. Torczon. Optimization using surrogate objectives on a helicopter test example, 1998.

[8] A. J. Booker, J. E. Dennis, Jr., P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset. A rigorous framewoerk for optimization of expensive functions by surrogates, November 1998.

[9] R. Christensen. *Linear Models for Multivariate, Time Series, and Spatial Data.* Springer-Verlag, 1991.

[10] A. Conn, K. Scheinberg, and P. Toint. On the convergence of derivative-free methods for unconstrained optimization. In *Approximation theory and optimization (Cambridge, 1996)*, pages 83–108. Cambridge Univ. Press, Cambridge, 1997.

[11] A. R. Conn, K. Scheinberg, and P. L. Toint. Recent progress in unconstrained nonlinear optimization without derivatives. *Math. Programming*, 79(1-3, Ser. B):397–414, 1997. Lectures on mathematical programming (ismp97) (Lausanne, 1997).

[12] A. R. Conn, K. Scheinberg, and P. L. Toint. A derivative free optimization algorithm in practice. In *Proceedings of 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, St. Louis, MO, 1998.

[13] D. D. Cox and S. John. SDO: a statistical method for global optimization. In M. Alexandrov and M. Hussaini, editors, *Multidisciplinary Design Optimization: State of the Art*, pages pp 315–29, Philadelphia, PA, 1997. SIAM.

[14] N. A. C. Cressie. *Statistics for Spatial Data.* Wiley-Interscience, 1993. Revised edition.

[15] C. Currin, T. Mitchell, M. Morris, and D. Ylvisaker. Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments. *Journal of the American Statistical Association*, 86(416):953–963, December 1991.

[16] J. Dagpunar. *Principles of Random Variate Generation.* Oxford Science Publications. Oxford University Press, 1988.

[17] P. W. Davis. Industrial-strength optimization at Boeing. *SIAM News*, 29(1), 1996.

[18] J. Dennis, D. Serafini, and V. J. Torczon. Optimization using surrogate objectives. Presented at the 16th International Symposium on Mathematical Programming, Lausanne, August 24–29, 1997.

[19] J. Dennis and V. J. Torczon. Managing approximation models in optimization. Technical Report CRPC-TR95550, Center for Research on Parallel Computing, Rice University, July 1995.

[20] H. Dym and H. P. McKean. *Fourier Series and Integrals.* Academic Press, 1972.

[21] Environmental Modeling Systems Inc. http://www.ems-i.com/gmshelp/Interpolation/Interpolation_Schemes/Kriging/Kriging.htm.

[22] R. Fletcher. *Practical Methods of Optimization.* John Wiley & Sons, 1991. second edition.

[23] C. A. Floudas. Deterministic global optimization, 2000.

[24] P. D. Frank. Optimization and analysis via surrogate modelling. Presented at the 16th International Symposium on Mathematical Programming, Lausanne, August 24–29, 1997.

[25] J. E. Gentle. *Random Number Generation and Monte Carlo Methods.* Springer, 1998.

[26] G. H. Golub and C. F. V. Loan. *Matrix Computations.* The Johns Hopkins University Press, 1989. second edition.

[27] G. R. Grimmett and D. R. Stirzaker. *Probability and Random Processes.* Oxford University Press, 1990.

[28] H.-M. Gutmann. A radial basis function method for global optimization. *Journal of Global Optimization*, 19:201–227, 2001.

[29] HSL. A collection of Fortran codes for large-scale scientific computation, 2002. See http://hsl.rl.ac.uk/.

[30] D. R. Jones. Global optimization with response surfaces. presented at the Fifth SIAM Conference on Optimization, Victoria, Canada, 1996.

[31] D. R. Jones. Black-box global optimization with nonlinear inequality constraints and no tuning parameters. Presented at the 16th International Symposium on Mathematical Programming, Lausanne, August 24–29, 1997.

[32] D. R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21:345–383, October 2001.

[33] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, October 1993.

[34] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13:455–492, 1998.

[35] D. R. Jones and W. J. Welch. Global optimization using statistical models. Technical report, General Motors, 1997.

[36] D. R. Jones, W. J. Welch, and M. Schonlau. Recent advances in Bayesian global optimization. Presented at the 16th International Symposium on Mathematical Programming, Lausanne, August 24–29, 1997.

[37] J. R. Koehler and A. B. Owen. Computer experiments. Technical Report, Stanford University, 1995.

[38] J. R. Koehler and A. B. Owen. Computer experiments. Handbook of Statistics, Vol.13, Elsevier Science B.V., 1996.

[39] T. G. Kolda, R. M. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM REVIEW*, 45(3):385–482, September 2003.

[40] M. J. Kushner. A versatile stochastic model of a function of unknown and time varying form. *Journal of Mathematical Analysis and Applications*, 1962.

[41] M. J. Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 8(6):97–106, 1964.

[42] M. Locatelli. Bayesian algorithms for one-dimensional global optimization. *Journal of Global Optimization*, 1997.

[43] M. Locatelli and F. Schoen. Random linkage: a family of acceptance/rejection algorithms for global optimisation. *Mathematical Programming*, 85(2):379–396, June 1999.

[44] S. N. Lophaven, H. B. Nielsen, and J. Søndergaard. Aspects of the matlab toolbox DACE. Technical Report IMM-REP-2002-12, Informatics and Mathematical Modelling, Technical University of Denmark, DK-2800 Kongens Lyngby - Denmark, 2002.

[45] S. N. Lophaven, H. B. Nielsen, and J. Søndergaard. DACE - a matlab kriging toolbox. Technical Report IMM-REP-2002-12, Informatics and Mathematical Modelling, Technical University of Denmark, DK-2800 Kongens Lyngby - Denmark, 2002.

[46] K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis.* Academic Press, London, 1979.

[47] J. Mockus. Application of Bayesian approach to numerical methods of global and stochastic optimization. *Journal of Global Optimization*, 1994.

[48] J. Mockus. Bayesian heuristic approach to global optimization and examples. *Journal of Global Optimization*, 2002.

[49] M. Mongeau, H. Karsenty, V. Rouzé, and J.-B. Hiriart-Urruty. Comparison of public-domain software for black box global optimization. *Optimization Methods & Software*, 13(3):203–226, 2000.

[50] M. D. Morris, T. J. Mitchell, and D. Ylvisaker. Bayesian design and analysis of computer experiments: Use of derivatives in surface prediction. *Technometrics*, 35(3):243–255, 1993.

[51] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization.* Wiley, 1988.

[52] Numerical Algorithms Group. Fortran 77 library mark 18, Oct 1994.

[53] P. M. Pardalos, H. E. Romeijn, and H. Tuy. Recent developments and trends in global optimization. *Journal of Computational and Applied Mathematics*, 124:209–228, 2000.

[54] Parzen. *Stochastic Processes.* Holden-Day Series in Probability and Statistics. Holden-Day, Inc., 728 Montgomery Street, San Francisco, California, 1965.

[55] M. J. D. Powell. A direct search optimization method that models the objective and constraint functions by linear interpolation. In *Advances in Optimization and Numerical Analysis, Proceedings of the Sixth Workshop on Optimization and Numerical Analysis,Oaxaca, Mexico*, volume 275, pages 51–67, Dordrecht, NL, 1994. Kluwer Academic Publishers.

[56] M. J. D. Powell. A direct search optimization method that models the objective by quadratic interpolation. Presentation at the 5th Stockholm Optimization Days, 1994.

[57] M. J. D. Powell. Direct search algorithms for optimization calculations. *Acta Numerica*, pages 287–336, 1998.

[58] H. A. Priestly. *Introduction to Integration.* Oxford Science Publications. Oxford University Press, 1997.

[59] K. F. Riley, M. P. Hobson, and S. J. Bence. *Mathematical Methods for Physics and Engineering.* Cambridge University Press, 2nd edition, 2002.

[60] J. Sacks, S. B. Schiller, and W. J.Welch. Designs for computer experiments. *Technometrics*, 31(1):41–47, 1989.

[61] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4:409–435, 1989.

[62] M. J. Sasena. *Flexibility and Efficiency Enhancements for Constrained Global Design Optimization with Kriging Approximations.* PhD thesis, University of Michigan, 2002.

[63] M. Schonlau. *Computer experiments and global optimization.* PhD thesis, University of Waterloo, 1997.

[64] M. Schonlau, W. J.Welch, and D. R. Jones. A data-analytic approach to Bayesian global optimization. In *American Statistical Association Proceedings, Section of Physical Engineering Sciences*, pages 186–191, 1997. .

[65] M. Schonlau, W. J. Welch, and D. R. Jones. Global versus local search in constrained optimization of computer models. Technical Report RR-97-11, Institute for Improvement in Quality and Productivity, University of Waterloo, 1997.

[66] M. Schonlau, W. J. Welch, and D. R. Jones. Global versus local search in constrained optimization of computer models. In N. Flournoy, W. F. Rosenberger, and W. K. Wong, editors, *New Developments and Applications in Experimental Design*, volume 34 of *IMS Lecture Notes- Monograph Series*, pages 11–25, Hayward, California, 1998. Institute of Mathematical Statistics.

[67] S. R. Searle. *Matrix Algebra Useful for Statistics.* John Wiley & Sons, 1982.

[68] G. A. F. Seber. *Multivariate Observations.* John Wiley & Sons, 1984.

[69] V. Torczon and W. Trosset. Sequential computer experiments for numerical optimization.

[70] V. Torczon and W. Trosset. Using approximations to accelerate engineering design optimization. In *Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 1998.

[71] A. Törn and A. Žilinskas. *Global Optimization.* Lecture Notes in Computer Science. Springer-Verlag, 1987.

[72] W. J. Welch, R. Buck, J. Sacks, H. P. Wynn, T. J. Mitchell, and M. D. Morris. Screening, predicting, and computer experiments. *Technometrics*, 34(1):15–25, 1992.

[73] D. Winfield. *Function and functional optimization by interpolation in data tables.* PhD thesis, Harvard University, 1969.

[74] A. Žilinskas. Axiomatic approach to statistical models and their use in multimodal optimization theory. *Mathematical Programming*, 22(1):104–116, January 1982.

[75] A. Žilinskas. Axiomatic characterization of a global optimization algorithm and investigation of its search strategy. *Operations Research Letters*, 4(1):35–39, May 1985.

[76] A. Žilinskas. A review of statistical models for global optimization. *Journal of Global Optimization*, 2:145–153, 1992.