

**SIMULATION METAMODELING AND OPTIMIZATION
WITH AN ADDITIVE GLOBAL AND LOCAL GAUSSIAN
PROCESS MODEL FOR STOCHASTIC SYSTEMS**

MENG QUN

(B.Eng., Shanghai Jiao Tong University)

**A THESIS SUBMITTED
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
DEPARTMENT OF INDUSTRIAL SYSTEMS
ENGINEERING AND MANAGEMENT
NATIONAL UNIVERSITY OF SINGAPORE**

2017

Supervisor:

Associate Professor Ng Szu Hui

Examiners:

Associate Professor Lee Loo Hay

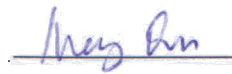
Associate Professor Ng Kien Ming

Professor Zelda Zabinsky, University of Washington

Declaration

I hereby declare that the thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

A handwritten signature in blue ink, appearing to read 'Meng Qun', is written over a horizontal line.

Meng Qun

20 January 2017

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Professor Ng Szu Hui for her invaluable guidance, support, and encouragement throughout my graduate study. Without her insights and constructive suggestions, I would not be able to complete this thesis.

I am deeply grateful to Professor Lee Loo Hay and Professor Ng Kien Ming for serving as my thesis committee members. I also acknowledge National University of Singapore for providing me with the research scholarship.

I also wish to express my gratitude to my research group mates, Wang Songhao, Li Guilin, Zhang Nan, Yuan Jun for their warm support and encouragement throughout my doctoral studies. Many thanks to Dr. Giulia Pedrielli, Dr Ji Yibo, Dr. Zhang Linmiao and Liu Weizhi for their kind help and guidance throughout the Ph.D. journey. My gratitude also goes to all my friends in Singapore. I spent enjoyable time in Singapore thanks to their companion.

Last but not least, I would like to thank my husband for his continuous support throughout these years, especially during those tough days.

Meng Qun, January, 2017

Contents

1	INTRODUCTION	1
1.1	Computer Simulation Models	1
1.2	Metamodels for Simulation Models	3
1.3	Simulation Optimization	5
1.4	Objectives and Scopes	8
1.5	Thesis Organization	10
2	LITERATURE REVIEW	12
2.1	Review of Metamodels	12
2.1.1	Polynomial Regression	13
2.1.2	Radial Basis Functions	13
2.1.3	Multivariate Adaptive Regression Splines	14
2.1.4	Gaussian Process Models	15
2.1.5	Artificial Neural Networks	16
2.2	Review of Gaussian Process Models for Large Data Sets	16
2.2.1	Global Approximation	17
2.2.2	Local Approximation	18
2.2.3	Combination of Global and Local Approximation	18
2.3	Metamodel-based Simulation Optimization Algorithms	19
2.3.1	Response Surface Methodology	20
2.3.2	Trust Region Methods	20
2.3.3	Efficient Global Optimization	21
2.3.4	Stochastic Response Surface Methods	22

3	AN ADDITIVE GLOBAL AND LOCAL GAUSSIAN PROCESS MODEL FOR LARGE DATA SETS	23
3.1	Introduction	23
3.2	Model Formulation	26
3.2.1	Stochastic Gaussian Process Model Basics	26
3.2.2	Overview of the AGLGP model	27
3.2.3	The Development of the AGLGP Model	28
3.2.4	Selection of Inducing Points and Local Regions	30
3.2.4.1	Partitioning the Design Space into Local Regions	32
3.2.4.2	Determining Inducing Points	32
3.3	Model Estimation	33
3.3.1	Estimating Predictive Distribution and Parameters in a One Stage Approach	34
3.3.1.1	Estimating Predictive Distribution in One Stage	34
3.3.1.2	Estimating Parameters in One Stage	36
3.3.2	Estimating the Predictive Distribution and Parameters with a Faster Two Stage Approach	39
3.4	Identifiability of the AGLGP model	42
3.5	Numerical Experiments	44
3.5.1	Effects of Inducing Points and Local Regions on AGLGP Model Estimation	44
3.5.2	Comparative Studies for the AGLGP Model	46
3.6	Conclusions	49
4	COMBINED GLOBAL AND LOCAL METHOD FOR STOCHASTIC SIMULATION OPTIMIZATION WITH AN AGLGP MODEL	50
4.1	Introduction	50
4.2	The Expected Improvement Function and the Modified Expected Improvement Function	53
4.3	Development of Methodology	54
4.3.1	General Framework of the CGLO Algorithm	54

4.3.2	Global Search Stage	56
4.3.2.1	Generation of Candidate Points	56
4.3.2.2	Global Expected Improvement	57
4.3.3	Local Search Stage	58
4.3.3.1	Local Search Step	59
4.3.3.2	Local Allocation Step	60
4.4	Convergence of the CGLO Algorithm	62
4.5	Numerical Results	66
4.5.1	One-dimensional Test Function (Illustration of Algorithm)	67
4.5.2	Comparative Studies with other Optimization Algorithms	69
4.6	A Navigational Safety Problem	71
4.7	Conclusions	75

5 PARALLEL GLOBAL AND LOCAL OPTIMIZATION WITH AGLGP

	MODEL	77
5.1	Introduction	77
5.2	Desired Properties of Parallel Search and Sampling Distributions	80
5.3	Basics and Background: Multi-point Expected Improvement and Pattern Search	84
5.3.1	Multi-point Expected Improvement	84
5.3.2	Pattern Search	85
5.4	Development of Methodology	86
5.4.1	General Framework of the PGLO Algorithm	86
5.4.2	Global Search Stage	87
5.4.2.1	Multi-point Global Expected Improvement	89
5.4.3	Parallel Local Search Stage	90
5.4.3.1	Selection of Initial Evaluation Points	90
5.4.3.2	Selection of Follow-up Evaluation Points	91
5.4.4	Allocation Stage	92
5.5	Convergence of the PGLO Algorithm	93
5.6	Numerical Studies	95

CONTENTS

5.6.1	Comparison with CGLO	95
5.6.2	Comparison with Other Parallel Pattern Search Techniques	98
5.7	A Navigational Safety Problem	103
5.8	Conclusions	106
6	CONCLUSIONS AND FUTURE RESEARCH	107
6.1	Summary	107
6.2	Future Research	109
	Bibliography	111
A	Derivation of Predictive Distribution in Section 3.3.1.1	118
B	Proof of Theorem 3.1	119
C	Proof of Proposition 3.2	121
D	Proof of Identifiability in Section 3.4	122
E	Test Functions in Section 5.6.2	124

Summary

Stochastic simulation models are widely used to provide an effective and efficient way to evaluate the behaviour of real systems. However with the stochastic and complex nature of most real systems, the simulation models can be time consuming to execute. Even when metamodels are developed to approximate the simulation models, estimating an appropriate metamodel can still be computationally challenging. This thesis proposes an additive global and local Gaussian Process model as a flexible surrogate for stochastic simulation models. This model attempts to capture the overall global spatial trend and the local trends of the responses separately, to enable more accurate modelling of the surfaces that are nonstationary in both the underlying function and the stochastic noise. The proposed additive structure of the model reduces the computational complexity in model fitting, and allows for more efficient predictions with large data sets. Based on the global and local structure of this model, we further integrate the model into a combined global and local simulation optimization algorithm and show the performance and properties of the algorithm. Furthermore, numerical results suggest that the proposed optimization framework can work more efficiently than other metamodel based optimization algorithms, especially when the search iteration progresses and the data size gets large. Finally, a parallel version of the optimization algorithm is developed to further reduce the computational time. A case study is presented to demonstrate the application of our approach in a navigational safety problem.

List of Tables

3.1	Error measurement of AGLGP model with different clustering techniques in local regions	45
3.2	Error measurement of AGLGP model with different region separation techniques	46
3.3	Error measurement of approximation models with one-dimension test function	48
3.4	Discontinuity measurement of AGLGP and LGP models with one-dimension functions Test 1 and Test 2	48
4.1	Overview of CGLO	55
4.2	Average performance with 5,000 and 10,000 simulation replications . .	71
4.3	The number of objective function evaluations and optimal probability of conflict $y_{approach}$ found by each optimization algorithm	74
4.4	The number of objective function evaluations and the deviation of the optimal probability of conflict $y_{approach}$ found by each optimization algorithm to the true optimal	75
5.1	Overview of PGLO	88
5.2	Performance of CGLO and PGLO per iteration	97
5.3	Average wall clock time to get a reasonable solution with a relative error < 1% using $q=1,4,8$ Processors	99
5.4	Relative speedup of parallel optimization algorithm when using $q=4,8$.	100
5.5	Average wall clock (W.C.) time and relative speedup to get a reasonable solution with a relative error < 1% using $q=1,4,8$ Processors (<i>small noise</i>)	101

5.6 Average wall clock (W.C.) time and relative speedup to get a reasonable solution with a relative error $< 1\%$ using $q=1,4,8$ Processors (*large noise*) 102

5.7 The number of objective function evaluations and the deviation of the optimal probability of conflict $y_{approach}$ found by each optimization algorithm to the true optimal 104

5.8 Number and percentage of scenarios for which each algorithm converges to the global optimal solution with 4 processors within 5 minutes 105

List of Figures

3.1	Plot of function $y(x) = \sin(30(x-0.9)^4) \cos(2(x-0.9)) + (x-0.9)/2$, the global, local and overall models	27
3.2	The inducing points and local regions	28
3.3	Inducing points and local regions	31
3.4	Test 1 function	47
3.5	Test 2 function	47
4.1	Initial fit	67
4.2	Iteration 1 of CGLO	68
4.3	Iteration 2 of CGLO	68
4.4	$g(x_1, x_2)$ function	69
4.5	Estimated optimal value of TSSO, EQI and CGLO with CPU time of 1400s.	70
4.6	Definition of Trajectory	72
4.7	Probability of conflict in (a) and log transformation of probability in (b)	73
5.1	AGLGP model fit with design point selected by modified Expected Im- provement	81
5.2	AGLGP model fit with design point selected by pattern search	82
5.3	mEI function and AGLGP model fit with design point selected by multi- start pattern search	83
5.4	Initial fit	95
5.5	Iteration 1 for CGLO and PGLO	96
5.6	Iteration 2 for CGLO and PGLO	97

List of Abbreviations

GP	Gaussian Process
AGLGP	Additive Global and Local Gaussian Process
MNEK	Modified Nugget Effect Model
CGLO	Reproducing Kernel Hilbert Space
EQI	Expected Quantile Improvement
TSSO	Two Stage Sequential Optimization
PGLO	Parallel Global and Local Optimization
OCBA	Optimal Computing and Budget Allocation
PS	Pattern Search
RS	Random Search
LOOCV	Leave-One-Out Cross Validation

Chapter 1

INTRODUCTION

This thesis contributes to the simulation metamodeling and optimization for stochastic systems. In this chapter, we first provide an overview of the thesis. The background and development of computer simulation models are introduced in Section 1.1. Section 1.2 introduces the development of metamodels. Section 1.3 briefly reviews the current progress of simulation optimization algorithms. The objectives and the scopes of this thesis are presented in Section 1.4 followed by the organization of this thesis in Section 1.5.

1.1 Computer Simulation Models

A computer simulation model is a computer program that attempts to simulate the behaviour of a specific complex system which cannot be modeled analytically or where analytic solutions are unavailable. The application of computer simulation models provides an efficient and effective way to study and analyze the characteristics of the complex systems in scientific, economics and engineering fields, such as the electronic circuit design problem in manufacturing industry (Currin et al., 1991), pricing the financial products problem in financial investment (Glasserman, 2003), and the planning of maintenance operations for airlines (Duffuaa and Andijani, 1999). The agent-based model is an advanced computer simulation model that simulates the actions and iterations of autonomous agents. It has become more popular in the last few years for real world

systems such as market systems and maritime transportation systems (Davidsson et al., 2005; Bonabeau, 2002).

Computer simulation models are commonly derived as simplifications of real systems as the computer experiments conducted on computer simulation models require a comparatively lower cost. The use of computer simulation models has several advantages over the direct analysis of real systems:

1. Computer simulation models are usually cheaper and easier to build compared with physical experiments. They are based on computer programs without simulating on real systems.
2. Computer simulation models can be used to recognize cause and effect relationships, identify the importance of different factors and predict the behaviour of a system at unknown conditions.

Various computer simulation models can be categorized in different ways based on the characteristics of the underlying systems, such as discrete or continuous models, static or dynamic models. Another common categorization method is to divide the computer simulation models as deterministic and stochastic simulation models. For a deterministic simulation model, simulation outputs are always the same for a specified input set. In contrast, a stochastic simulation model contains randomness in the outputs to represent uncontrollable factors just like real systems.

Deterministic simulation models have been widely used in practice due to their convenience, especially when we are interested in the average behavior of a system or the randomness has low impact on the system's performance. It requires comparatively lower cost to obtain the results given a set of inputs. Examples can be found in various areas such as Computer Aided Engineering (CAE) and Computer Aided Design (CAD) (see Kleijnen (2008) and Santner et al. (2003)).

Different from deterministic simulation models, stochastic simulation models include randomness in the outputs to represent the stochastic nature of real systems. For example, some uncontrollable factors like weather and fluctuation, can bring randomness in the response, while in a queueing system, the arrival rate and the service time are all random. Hence, a single simulation run is no longer sufficient for a specific in-

put because the stochastic simulation model will deliver different outputs over different replications for a specific input.

1.2 Metamodels for Simulation Models

Although simulation models are widely used to provide an effective and efficient way to evaluate the behaviour of real systems, due to the complex nature of most real systems, the simulation models can be time consuming to execute. The computational cost of running expensive simulation models becomes a critical issue.

To reduce the computational cost of running expensive simulation models, one common simplification is to develop metamodels (also known as surrogates or response surface models) to approximate the outputs of simulation models. A metamodel is a statistical model of a simulation model with a closed mathematical form that emulates the behaviour of the simulation model. Examples of metamodels include polynomial regression models, Gaussian Process models (also known as kriging), radial basis functions (RBF), multivariate adaptive regression splines (MARS), artificial neural networks (ANN), and support vector regression (SVR) models. Simpson et al. (2001) reviewed the application of metamodels in engineering systems. Li et al. (2010) also provided a comprehensive comparison of metamodels in simulation optimization. Among all these metamodels, Gaussian Process models (also known as kriging models) (Cressie, 1993), were first introduced into the field of design of experiments by Sacks et al. (1989) and have become popular in recent years due to its adaptability and efficiency in modeling simulation outputs (Santner et al., 2003). It has also been well applied in the field of pattern recognition and machine learning (Bishop, 2006; Rasmussen and Williams, 2006).

The use of Gaussian Process models is also a popular technique to solve and analyse stochastic simulation models in recent years. Different from deterministic simulation models, the randomness and complexity of stochastic simulation models raise another critical issue that a single simulation run is no longer sufficient for a specified input set. More simulation replications are required to estimate the expectation of stochastic simulation outputs. Hence, stochastic simulation models require much more compu-

tational time to analyse than deterministic simulation models, and the application of Gaussian Process models essentially helps to understand the behavior of stochastic simulation models. Some of the Gaussian Process models for deterministic simulation models (Sacks et al., 1989; Santner et al., 2003) have been successfully extended in stochastic situations. More specifically speaking, because stochastic simulation models can be divided into two different scenarios: homoscedastic case (with a random noise that is assumed to be Normally, Independently and Identically distributed (NIID)) and heteroscedastic case (with a random noise whose variance changes over the domain space), metamodels have been developed to handle those different scenarios. For example, the nugget effect model with homoscedastic assumption can perform well in the homoscedastic case (Cressie, 1993), while the stochastic Gaussian Process model has been shown to perform well in the heteroscedastic case (Ankenman et al., 2010; Yin et al., 2011).

Estimating the stochastic Gaussian Process model, however, is a computational challenge when the data sets are large. In simulations, large datasets may be required to analyse a nonstationary simulation model whose outputs can change dramatically over the whole space. In optimization problems, the number of evaluation points can increase quickly as the optimization algorithm progresses towards the optimal solution. Given the data size of N , estimating model parameters using traditional methods like maximum likelihood estimation and estimating the model predictors involve the inversion of a $N \times N$ covariance matrix, which typically requires $O(N^3)$ operations and $O(N^2)$ memory. This becomes computationally intractable when N is large. Hence, approximation techniques are needed to apply the Gaussian Process model for large data sets (Sang and Huang, 2012; Snelson and Ghahramani, 2007).

Gaussian Process models have also been applied to analyse the highly nonstationary simulation models whose simulation outputs have different degrees of smoothness in one region than another; Gaussian Process models with various covariance structures enable more flexibility in modelling the dramatic changes over the whole space (Rasmussen and Williams, 2006). This non-stationarity is common in engineering systems, and can be caused by the heteroscedastic noise of stochastic systems or a highly non-linear response surface. The stochastic Gaussian Process model (Ankenman et al., 2010;

Yin et al., 2011) is able to capture the nonstationarity in the heteroscedastic noise. A highly non-linear response surface can be found in production planning, when different models for different components are integrated to generate finished goods. Hence, system behaviors can differ significantly in distinctive design regions. Similarly in maritime transportation, one measure of safety for a vessel at sea is the probability of encountering a conflict. This measure can change drastically with the angle of turn in heavy traffic regions. Many approaches have been proposed to address these nonstationary problems. For example, Ba and Joseph (2012) extended the Gaussian Process model with a composite covariance structure to capture the nonstationarity in the simulation model. The nonstationarity can also be addressed by partitioning the input space into regions and fitting separate independent stationary Gaussian Process models in each region (Kim et al., 2005). The input space can also be partitioned by a more sophisticated Bayesian treed structure (Gramacy and Lee, 2008). However, these approaches cannot be well applied for stochastic simulations with heteroscedastic random noise as the random variability of stochastic responses can considerably affect the estimation of the underlying deterministic response surfaces.

1.3 Simulation Optimization

Computer simulation models are not only helpful in studying the characteristics of complex systems, but also widely adopted to solve optimization problems. Instead of analytically deriving the optimal solution, multiple solutions generated by optimization algorithms can be iteratively evaluated on computer simulation models. The combination of simulation models and optimization algorithms is referred as simulation optimization in the literature. This approach has been proven to be particularly effective for complex functions, especially for black box functions. In general, decision makers target to find a set of parameters that yield the optimal performance. The general global optimization problem can be expressed as

$$\min_{x \in \mathcal{X}} f(x) \quad (1.1)$$

where $f : \mathcal{X} \rightarrow \mathcal{Y} \subseteq \mathbb{R}$ is the deterministic objective function, and $\mathcal{X} \subseteq \mathbb{R}^d$ is a compact feasible region in \mathbb{R}^d . The objective function f typically has no closed form and can

only be evaluated through an expensive and complex black-box simulation model. In this thesis, we focus on stochastic simulation models, where $f(x)$ cannot be obtained directly, but rather sample observations with noise $y(x)$ can be observed. Hence, we are interested in solving the following problem:

$$\min_{x \in \mathcal{X}} E(y(x)) \tag{1.2}$$

Simulation optimization strategies can be divided into several categories based on the nature of the response f and the input space \mathcal{X} . If the input space is discrete, appropriate optimization methods include direct search methods like random search and meta-heuristics. In particular when the input space is finite and small, ranking and selection is a promising strategy. On the other hand, if the input space is continuous, gradient-based methods and metamodel-based optimization methods can be applied (Barton and Meckesheimer, 2006).

Direct search methods call the simulator at each iteration to obtain an estimate of the response and the search is also conducted on the simulator. Popular direct search methods include random search and its adaptations such as COMPASS (Hong and Nelson, 2006; Xu et al., 2010), nested partition methods (Shi and Ólafsson, 2000), pattern search and its extension (Torczon, 1997), and other heuristic methods such as genetic algorithms and simulated annealing approaches. Some direct search methods have been shown to be globally convergent. A major drawback of this family of approaches, however, is their cost when simulation runs require high computational effort. In such cases, metamodeling based methods offer the possibility to use the information from simulation runs to get insight about regions where simulations have not been performed yet.

If the input space is continuous, other search methods can be applied. Stochastic gradient-based optimization methods such as stochastic approximation can use efficient methods such as likelihood ratios, or less efficient finite-difference approximations, for estimating the gradient of f . These methods effectively search the input space for optimal solution without attempting to provide a global approximation of the response surface f . However, it can fail on stochastic responses with large noise variability or when the gradient information is computationally expensive.

Another sensible approach for gradient-free optimization is the metamodel-based methods. They estimate a metamodel with few simulations in the search procedure, to quickly predict the performance at any given point in the domain space without the need to run the simulator at every potential point. Such methods provide the information of the entire surface to better identify the points for further simulations and locate the optimum. Barton and Meckesheimer (2006) provided an overview of metamodel types and the overall strategy of metamodel-based optimization methods. Response Surface Methodology (RSM) (Kleijnen et al., 2004) is one of the most popular techniques in this class for its ease of implementation. RSM sequentially explores small regions with a first order and a second order linear regression model for new experimental designs.

In applying the Gaussian Process model as a surrogate for optimizing objective functions, a sequential approach is typically applied. Jones et al. (1998) proposed a sequential optimization method based on the Gaussian Process model. The proposed Expected Improvement (EI) function and the Efficient Global Optimization (EGO) algorithm balance the trade-off between exploration (searching the whole space for regions that have not been explored before) and exploitation (searching around the current optimal solution for better solutions) for the optimum of the deterministic simulation model. Huang et al. (2006) extended the EGO scheme for stochastic simulation models with homogeneous variance throughout the whole space, and proposed the Sequential Kriging Optimization (SKO) method. With the nugget effect model and the augmented EI function, the SKO method accounts for the influence of random noises, and considers selecting and adding replications on the existing evaluated points other than searching for new points. Picheny et al. (2013) and Quan et al. (2013) further extended EGO to heteroscedastic case where random noises are assumed to have nonconstant variance. The proposed Expected Quantile Improvement (EQI) by Picheny et al. (2013) is a more general quantile-based criterion that accounts for the users risk tolerance. EQI considers the variance of the noise at un-evaluated locations when searching for a new point, but it also requires the noise variance function to be known. The two-stage sequential optimization (TSSO) algorithm (Quan et al., 2013) relaxes the requirement of known variance. TSSO comprises of a search stage to determine the next evaluation point and an allocation stage to evaluate the best optimal solution by running simulations at each eval-

uated point according to the Optimal Computing Budget Allocation (OCBA) strategy (Chen and Lee, 2010). By ignoring predictor uncertainty caused by random variability, the search stage can focus on new points with low predicted response or high spatial uncertainty, while the search stage with a minimum number of replications gives insight for the noise variance at the new point.

As the originally proposed EI function and EGO algorithm are designed based on Gaussian Process models, which scale poorly with the number of data points, They can only work efficiently with a small limited computing budget. For a high nonstationary response surface, EGO might require a sufficient amount of replications to get a reasonably good solution. In this situation, EGO may not work efficiently any more as the searching procedure (including refitting of the Gaussian Process models) becomes expensive when the number of evaluations observed gets large as the iteration progresses to find the optimum. Hence, it motivates the derivation of a more efficient approach applicable for more general simulation models. Other than reducing the computational complexity for model estimation with certain approximation techniques, the algorithm should also consider reducing the iterative refitting of Gaussian Process models.

1.4 Objectives and Scopes

As discussed in the previous sections, there are still several research gaps in the field of simulation metamodeling and optimization, some of which can be summarized as follows,

- Although Gaussian Process models have been shown to be a very useful meta-model form to approximate computer simulation models, there still exists limited work on the approximation of nonstationary Gaussian Process models in stochastic systems. The existing ones either consider only the nonstationarity in the deterministic function or consider only the nonstationarity in the heteroscedastic noise. This form will be useful as many computer simulation models of complex systems today are highly complicated, and fast approximations of them are required to facilitate real time decisions on those systems.

1.4. OBJECTIVES AND SCOPES

- The existing approximation methods of Gaussian Process models for large data sets are generally designed with a predefined covariance structure. They may not sufficiently consider the nonstationarity, or the approximation of a complex nonstationary covariance structure is more computationally complex.
- Metamodel-based global optimization algorithms seldom consider the computational complexity in the searching criterion. As the iteration progresses towards the global optimum, the data size gets larger, and the searching criterion based on the metamodel can become more expensive, especially for Gaussian Process model based optimization algorithms.

In light of these research gaps, this thesis aims at (i) developing a fast estimated metamodel that can be applied with large data sets and able to capture the nonstationarity for the underlying simulation models, and (ii) developing a metamodel-based global optimization strategy that can be applied for extensive types of simulation models. More specifically, the objectives of this thesis are to:

- Develop a novel additive model as an approximation of the stochastic kriging model for faster estimation and prediction with large data sets. It also provides the flexibility to capture the nonstationarity in the underlying simulation models.
- Provide statistical properties of the additive model, and compare analytically and numerically the performance of the predictions with different approximation models.
- Develop optimization algorithms for more general stochastic simulation models with nonstationary response surfaces. Both the sequential and parallel optimization strategies are considered.

The results of this work can provide some insights and improvements to the simulation metamodeling and optimization in a stochastic environment. More specifically, this work helps in improving

- The efficiency of the Gaussian Process model with large data sets.

- The effectiveness of the Gaussian Process model for nonstationary response surfaces.
- The performance and efficiency of the Gaussian Process model based optimization for stochastic simulation models.

A common issue for Gaussian process models is their poor performance when dealing with high dimensional variables, as the high dimensional variables will significantly increase the complexity of estimation for the sensitivity parameter for each dimension. In this thesis, however, we focus on metamodeling schemes for large data sets and their applications in optimization, for problems where the input dimensions are less than ten. As many applications in the engineering systems (e.g. 4 dimensional welded beam design (Rao, 1996), 3 dimensional helical compression spring (Arora, 2004) and 4 dimensional S^2TA system (Pedrielli et al.)) have limited number of decision variables, we only focus on these lower dimensional problems in this study.

1.5 Thesis Organization

This thesis contains six chapters. Chapter 2 gives a review of the metamodels and metamodel based optimization methods.

In Chapter 3, the additive global and local Gaussian Process model is proposed as the solution to general stochastic simulations with large data sets. We develop the model on the basis of the stochastic kriging model by dividing the whole stochastic process into a global model with a small set of inducing points to capture the global trend and piecewise independent local models to capture the residual process from the global model. We further allow different covariance structures for the local models to capture the nonstationarity across the whole space. We also propose an approach to determine the local regions, which has not been previously addressed but assumed given. Moreover, we show several nice properties of the additive model, such as identifiability. Finally, numerical studies are conducted to compare the performance of the proposed additive global and local Gaussian Process model and other existing approximation methods for large data sets.

In Chapter 4, we further develop a simulation optimization algorithm that leverages on the global and local structure of the additive model in Chapter 3 for a combined iterative global and local search. The proposed combined global and local optimization approach shares a similar framework with the Efficient Global Optimization (EGO), but works more efficiently with a global search stage that quickly narrows down the whole space into a promising local region and a local search stage that exploits within the promising local region for an optimal solution. We also propose a global expected improvement function in the global search stage to better account for the global trend (through the global model) and the global distribution of observations (through a small set of inducing points representing similar observations around). We then derive an allocation strategy that intelligently allocates budget to the evaluated points for the purpose of improving the metamodel fit and estimating the optimal solution. We analyze the global convergence property of the combined approach and study its performance on a test function and a practical navigational safety problem.

In Chapter 5, we further extend the combined global and local optimization algorithm developed in Chapter 4 in a parallel environment. The parallel framework includes a global search stage exploring the whole space for multiple promising local regions through a multi-point global expected improvement function and a parallel local search stage that selects multiple distinct points in each promising local region for simultaneous evaluations. We then incorporate locally convergent direct search methods for fast exploitation around each of those selected points. The performance of the combined global and local optimization algorithm and the parallel framework is demonstrated on a simple one-dimensional example. The efficiency of the parallel framework is further studied on 5 different test functions and the navigational safety problem.

Chapter 6 summarizes this work of the additive global and local Gaussian Process model in simulation metamodeling and optimization and provides some directions for future research.

Chapter 2

LITERATURE REVIEW

This chapter introduces simulation metamodeling and metamodel-based simulation optimization. Section 2.1 briefly reviews five commonly used metamodels, and we then focus on the more promising Gaussian Process model which is the model proposed to be applied in the following studies, and conduct a more comprehensive review of approximate computation methods for Gaussian Process models with large data sets in Section 2.2. Finally, in Section 2.3, we look into several metamodel based approaches for global optimization of black-box problems.

2.1 Review of Metamodels

Metamodels are built based on the data collected from target black-box simulation models to imitate their behaviours. For deterministic simulation models $f(x)$ with the simulation input x and the simulation output $y = f(x)$, the metamodel can be mathematically expressed by

$$\hat{f}(\sim) = \hat{f}_{\theta}(\sim) \quad (2.1)$$

as an approximation of the simulation output, where $\hat{f}(\sim)$ is the metamodel and θ is the set of parameters for the metamodel (omitted for simple expression), which can be a function of the inputs \mathbf{x} and the observed outputs \mathbf{y} . We Given a set of input x_0 , $\hat{f}(x_0)$ is the output of the metamodel, as a prediction for the simulation model's output $y_0 = f(x_0)$.

For stochastic simulation models where the simulation output is no longer deterministic at a given input, multiple simulation replications $y_j, j = 1, \dots, n$ at a given input x are averaged $\bar{y} = \sum_{j=1}^n y_j/n$ to estimate the expectation of the stochastic simulation outputs. Metamodels are then developed to approximate the expectation of stochastic simulation outputs.

2.1.1 Polynomial Regression

Polynomial regression model is the most popular and simplest metamodel that models the relationship between inputs and outputs as a n th polynomial. Specifically with the inputs $\{x_i\}_{i=1}^n$ and the outputs $\{y_i\}_{i=1}^n$, the general form of a n th degree polynomial for one independent variable is

$$\hat{f}(x) = \beta_0 + \sum_{i=1}^n \beta_i x^i \quad (2.2)$$

where β_i is the least square coefficient selected by minimizing the mean square error. We denote \mathbf{X} as the matrix where the ij th element is the i th input with j degree polynomial, i.e. $(x_i)^j$, and the coefficients are given as

$$\boldsymbol{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \quad (2.3)$$

Polynomial regression model has been well applied in the simulation context (Kleijnen, 1998). Ruppert (2011) has applied the polynomial regression model in the risk analysis and mutual fund evaluation in financial engineering. However, the polynomial regression model describes the simulation model behaviours with one simple function, which may show inadequacy in terms of the prediction accuracy, especially when the complex systems have local behaviours that vary from region to region.

2.1.2 Radial Basis Functions

The standard Radial basis functions (RBF) interpolation method was first proposed by Hardy (1971) for interpolation of scattered data, and was studied in more detail by

Buhmann (2003). The model uses a linear combination of radially symmetric functions. With evaluated points $\{x_i, y_i\}_{i=1}^n$, the RBF predictor takes the form

$$\hat{f}(x) = \sum_{i=1}^n \beta_i \phi(\|x - x_i\|) + p(x) \quad (2.4)$$

where the weights $\{\beta_i\}_{i=1}^n$ are determined by the interpolation $\hat{f}(x_i) = y_i$. $p(x)$ is polynomial tail that depends on the choice of ϕ . The basis function ϕ can take multiple forms: multiquadrics, thin plate splines, cubic splines, Gaussian, and inverse multiquadrics.

RBF models have been extensively studied due to their applicability in almost any dimension. A comprehensive study of RBF models for simulation metamodeling is conducted by Hussain et al. (2002). As a mesh-less technique, the RBF model was used in the numerical simulation related with Partial Differential Equation (PDE) (Kansa, 1990; Larsson and Fornberg, 2003).

2.1.3 Multivariate Adaptive Regression Splines

The multivariate adaptive regression splines (MARS) (Friedman, 1991) is an expansion of simple splines model. It is derived for flexible regression modeling of high dimensional data. With no assumption for the underlying relationship between inputs and outputs, the MARS model approximates simulation models by a forward stepwise algorithm to select splines for the model followed by a backward procedure to prune the model. The mathematical form can be written as:

$$\hat{f}(x) = \beta_0 + \sum_{k=1}^M \beta_k B_k(x) \quad (2.5)$$

where β_k is the coefficient and $B_k(x)$ is the basis function that is represented as

$$B_m(x) = \prod_{k=1}^{L_m} [S_{k,m}(x_{v(k,m)} - t_{k,m})] \quad (2.6)$$

Here the training data are divided into M separate regions. The basis function $B_m(x)$ is a combination of L_m functions fitted in region m . $S_{k,m} = \pm 1$ and $t_{k,m}$ is the knot value which is defined as the endpoints of regions. $x_{v(k,m)}$ is the v th input variable. As the algorithm goes forward, it updates with the truncated linear function involving a new

variable until the predefined upper limit on the number of basis functions is reached. The backward process prunes the basis functions based on their contributions. MARS is thoroughly compared with other metamodels in Jin et al. (2001) and Clarke et al. (2005) in simulation applications.

2.1.4 Gaussian Process Models

Gaussian Process model is firstly known as *Kriging* in geostatistics field, and then adopted for prediction in spatial statistics (Cressie, 1993) and experimental designs (Sacks et al., 1989; Santner et al., 2003). It is reasonable to assume the simulation outputs $y_i = f(x_i)$ and $y_j = f(x_j)$ are similar if x_i and x_j are close to each other to get a smooth metamodel. So it assumes all points in the domain space are spatially correlated, following a multivariate normal distribution. For example, the simulation outputs y_i and y_j follow the distribution

$$\begin{bmatrix} y_i \\ y_j \end{bmatrix} \sim N \left(\begin{bmatrix} \mu_i \\ \mu_j \end{bmatrix}, \begin{bmatrix} \sigma^2 & \sigma^2 R(y_i, y_j) \\ \sigma^2 R(y_j, y_i) & \sigma^2 \end{bmatrix} \right) \quad (2.7)$$

The correlation function $R(y_i, y_j)$ can take various forms, measuring the degree of closeness between x_i and x_j . For instance, the most commonly used exponential family of correlation functions for d -dimensional inputs take the form

$$R(y_i, y_j) = \exp\left(\sum_{h=1}^d \theta_h (x_{i,h} - x_{j,h})^p\right) \quad (2.8)$$

As seen in Equation (2.8), the correlation is evaluated only based on the distance between two input variables. The Gaussian Process model builds the predictor as a linear combination of observations,

$$\hat{f}(x) = \sum_{i=1}^n \lambda_i y_i, \quad \sum_{i=1}^n \lambda_i = 1 \quad (2.9)$$

where λ_i is a function of the n observed points x_1, \dots, x_n , their observations y_1, \dots, y_n and the correlation function $R(\cdot, \cdot)$, and is chosen to give a best linear unbiased predictor. The Gaussian Process model also provides a unique view of prediction uncertainty with

mean square error $\hat{s}^2(x)$ that is widely used in experimental design (Sacks et al., 1989) and simulation optimization (Jones et al., 1998), where

$$\hat{s}^2(x) = \sigma^2 - \mathbf{r}'\mathbf{R}^{-1}\mathbf{r}. \quad (2.10)$$

Here, $\mathbf{r} = (R(f(x), y_1), R(f(x), y_2), \dots, R(f(x), y_n))$ and \mathbf{R} is $n \times n$ correlation matrix with (ij) th item $R(y_i, y_j)$. However, the Gaussian Process model scales poorly with the number of observations. More detailed review of its approximation schemes for large data sets will be given in Section 2.2.

2.1.5 Artificial Neural Networks

The structure of the artificial neural networks (ANN) typically comprises of three layers: input layer, hidden layer, and output layer. Inspired from the mechanism of human nerve systems, each node on a layer is one neuron, and information is transformed between layers through the connection of neurons. The general form for ANN model with d -dimension input neurons $x = \{x_1, \dots, x_d\}$ and one dimension output neuron follows the form:

$$\hat{f}(x) = \sum_{i=1}^H w_i h \left(\sum_{j=1}^d v_{ij} h(x_j) + \alpha_i \right) + \beta \quad (2.11)$$

where $h(\cdot)$ is the transform function, and H is the total number of hidden neurons. α_i is the bias in i th hidden neuron, and β is the bias of the output neuron. ANN has been commonly used in multi-disciplinary research fields. Kilmer et al. (1997) initiated ANN as a metamodel for discrete stochastic simulation, and then it has been widely applied in simulation metamodeling (Nasereddin and Mollaghasemi, 1999; Fonseca and Navarrese, 2002; Fonseca et al., 2003).

2.2 Review of Gaussian Process Models for Large Data Sets

The previous studies of Gaussian Process models focus on the accuracy of approximating simulation models. However the model predictor in Equation (2.9) can become computationally intractable with a large number of observations $y_1 \dots, y_n$. This is because the

calculation of λ_i involves the inversion of a $n \times n$ covariance matrix \mathbf{R} (with each ij th item $R(y_i, y_j)$ representing the correlation between y_i and y_j), which typically requires $O(n^3)$ operations and $O(n^2)$ memory and can become computationally intractable for a large n . This has generated considerable interest in developing approximation models for large data sets. Rasmussen and Williams (2006) provides a thorough review of approximation methods. Approximations are made either to the GP regression with fixed parameters or to the marginal likelihood for parameter estimation. Generally, the existing approximation methods may be divided into three categories: global approximation, localized regression, and combination of global approximation and localized regression. We will separately review these three techniques in the following subsections.

2.2.1 Global Approximation

Global approximation methods include *rank reduction* and *sparse approximation* (Quiñonero Candela and Rasmussen, 2005; Banerjee et al., 2008). The rank reduction approximates the process by taking the leading terms in its *Karhunen-Loève* (*KL*) expansion (Baker, 1977). Keeping only the first m terms will give a rank m approximation. The simplest sparse approximation method is taking a subset of data as an approximation. This is not a competitive method as it does not fully consider the spatial uncertainty with only a part of the evaluated points. An alternative sparse approximation scheme is to derive a predictive process conditional on a latent process over a small set of *knots* (may or may not form a subset of the evaluated points). Both the rank reduction and the sparse approximation result in an approximation of the original covariance function $R(x, x')$,

$$\tilde{R}(x, x') = R(x, X^*)\mathbf{R}^{*-1}R(x', X^*) \quad (2.12)$$

where X^* is the set of knots, \mathbf{R}^* is the covariance matrix of X^* . However, these global approximation methods typically capture only the long lengthscale global trend of spatial processes, leaving much of local dependencies unexplained.

2.2.2 Local Approximation

The second category is localized regression, where model predictions are estimated based on local neighborhoods. Local Kriging fits different Kriging models in different subregions independently. Local Kriging is known for its adaptability to model nonstationary process and its efficiency in computation. However, it suffers from discontinuities at boundaries due to its localized independent model estimation. Park et al. (2011) proposed an approach that smooths the discontinuities by adding equality constraints at the boundaries of neighboring subregions, but additional computational time is required to estimate the values at boundaries. Another local approximation approach is to apply the covariance tapering, which assumes that distant pairs of observations are uncorrelated (Furrer et al., 2006). Let $R(\|x - x^*\|)$ denote the original covariance function. Consider a tapering function $K_{taper}(\|x - x^*\|, \lambda)$, which is an isotropic correlation function that is identically 0 whenever $\|x - x^*\| \geq \lambda$. The tapered covariance function is defined as

$$R_{taper}(\|x - x^*\|) = R(\|x - x^*\|)K_{taper}(\|x - x^*\|, \lambda)$$

Sparse matrix algorithm can then be applied to realize the computational efficiency. However, such approaches are unable to effectively capture the long lengthscale dependencies, often missing the larger global trend. Recent works by Gramacy and Apley (2014) and Gramacy and Haaland (2016) propose splitting the input domain into different segments where the parameters are estimated separately, enabling parallelization in the model estimation. However accomplishing the massive parallelization may still require large amounts of computation.

2.2.3 Combination of Global and Local Approximation

The last category combines the global approximation and the localized regression to overcome the disadvantages of each individual method. A full scale approximation (FSA) of covariance functions proposed by Sang and Huang (2012) approximates covariance functions through a combination of a reduced rank approximation and a tapered residual

approximation.

$$\tilde{R}(x, x') = Q(x, x') + (R(x, x') - Q(x, x')) \cdot K_{tapper}(x, x'; \gamma) \quad (2.13)$$

where $Q(x, x') = R(x, X^*)\mathbf{R}^{*-1}R(x', X^*)$ and $K_{tapper}(x, x'; \gamma)$ is a tapering function that decreases as $\|x - x'\|$ increases and is identically 0 when $\|x - x'\| \geq \gamma$. This attempts to capture both the long lengthscale dependence and shorter lengthscale dependence. As the local adjustment of FSA only captures the residual correlation of neighboring points, it performs well in estimating smooth responses with strong correlation between points. The partially independent conditional approximation (PIC) approach (Snelson and Ghahramani, 2007) also combines a reduced rank approximation and a locally independent residual approximation.

$$\tilde{R}(x, x') = Q(x, x') + \phi(x, x')(R(x, x') - Q(x, x')),$$

$$\text{where } \phi(x, x') = \begin{cases} 1 & \text{if } x, x' \in \text{same local region} \\ 0 & \text{otherwise} \end{cases}$$

The local adjustment of PIC is able to capture the correlation between points in the same local region. The local adjustment assumed in these approaches is however still restricted by the tapering factor and the global sensitivity parameter that controls the correlations between the global points, as the local correlations are controlled by functions of these parameters only. This can hinder its ability to capture the sharp changes in a high nonstationary response.

2.3 Metamodel-based Simulation Optimization Algorithms

Metamodel can be good alternatives to simulation models to approximate the relationship between inputs and outputs for black-box systems. Hence, a popular optimization method for continuous functions is based on metamodels. With an appropriate metamodel, we have an overview of the characteristics of black-box systems, which can be applied to identify promising points and guide search directions. In the following sub-

sections, several optimization algorithms that incorporate metamodels will be briefly introduced.

2.3.1 Response Surface Methodology

Response Surface Methodology (RSM) developed by Box et al. (1987) has been effectively used in many disciplines. In this approach, polynomial models are applied to approximate simulation models. The strategy of RSM is to sequentially explore local regions by fitting a lower order polynomial model, deciding the search direction following the steepest ascent search method, and fitting a higher order polynomial model when the search is close to the optimal solution. First-order and second-order polynomial models are preferred in RSM due to their simplicity and efficiency,

$$f_1(x) = \beta_0 + \sum_i^d \beta_i x_i$$

$$f_2(x) = \beta_0 + \sum_i^d \beta_i x_i + \sum_i^d \beta_{ii} x_i^2 + \sum_i^d \sum_{i < j}^d \beta_{ij} x_i x_j$$

where $x \in \mathbb{R}^d$, $x = \{x_i\}_{i=1}^d$. However polynomial models may not be able to provide an accurate global approximation. Due to the correlation between the high-order items and the lower-order items, the coefficient matrix might get ill-conditioned.

2.3.2 Trust Region Methods

Trust region methods, also known as restricted step methods, were first proposed by Celis et al. (1985). The traditional trust-region method builds a quadratic model to approximate the true function within a trust region.

$$\hat{f}(x + d) = f(x) + d' \nabla f(x) + \frac{1}{2} d' \mathbf{H} d \quad (2.14)$$

where d is the step size, $\nabla f(x)$ is the gradient of $f(x)$, and \mathbf{H} is the Hessian matrix of $f(x)$. In general, quadratic models only fit well in the neighborhood of x , which is defined as a trust region. If the estimation is adequate, the trust region increases,

otherwise the trust region decreases. The algorithm will solve the minimization of the quadratic model within the trust region.

The trust region methods can be modified with other metamodels. Conn et al. (1997) and Powell (2002, 2003) utilized the multivariate polynomial models in the trust region framework. It is also incorporated with kriging model (Gano et al., 2006) and radial basis functions (Wild et al., 2008). However, when applied with other metamodels where the gradient information is not available, other informatics functions (like the trust ratio function (Gano et al., 2006)) have to guide the search. Besides, the trust region method is designed for unconstrained local optimization, which might get complicated when tasked with constrained optimization or global optimization.

2.3.3 Efficient Global Optimization

Jones et al. (1998) developed the well-known Efficient Global Optimization (EGO) that sequentially updates the kriging model with the point that maximizes the Expected Improvement (EI) function. The general procedure is summarized as follows,

1. Run simulations at space-filling initial designs and build an initial kriging model.
2. Use cross validation to make sure the kriging model is satisfactory.
3. Find the next evaluation point that maximizes the EI function. If the maximal EI is sufficiently small, stop.
4. Run simulations at the new selected evaluation point and update the kriging model based on all evaluated points. And go back to Step 3.

The critical criterion in the EGO framework is the EI function, which evaluates both the probability of improvement based on the current optimal solution and the amount of possible improvement. The EI function is statistically formulated as

$$\begin{aligned} E[I(x)] &= E[\max[f_{min} - f(x), 0]] \\ &= (f_{min} - \hat{f}(x))\Phi\left(\frac{f_{min} - \hat{f}(x)}{\hat{s}}\right) + \hat{s}\phi\left(\frac{f_{min} - \hat{f}(x)}{\hat{s}}\right) \end{aligned}$$

where $f(x)$ is a random variable that satisfies a normal distribution $N(\hat{f}(x), \hat{s}^2(x))$, and $\phi(\cdot)$ and $\Phi(\cdot)$ are the standard normal density and distribution functions respectively.

EGO has earned extensive studies in recent years due to its promising performance. Sasena et al. (2002) claimed that EGO is sufficient and practical in engineering design problems. Sóbester et al. (2005) proposed a weighted expected improvement for a better control of the balance between exploration and exploitation. Kleijnen et al. (2012) improved the kriging predictor variance through bootstrapping and extended EGO with bootstrapped EI. However only limited research has investigated the performance of EGO in stochastic simulations, especially with heterogeneous variance. In this research, we also focus on the performance of EGO framework in stochastic situations.

2.3.4 Stochastic Response Surface Methods

The Stochastic Response Surface (SRS) framework was introduced by Regis and Shoemaker (2007) that simplifies the optimization problem of finding the next simulation point by generating random candidate points. SRS follows a similar framework with EGO. Different from EGO, SRS efficiently selects next evaluation points only among a set of candidate points without evaluating all possible solutions, and it is applicable for various metamodels and search criteria. The framework for SRS is shown below.

1. Do function evaluations at a set of space-filling initial designs.
2. Fit/update the response surface model.
3. Randomly generate candidate points
4. Select the next function evaluation point among the *candidate points*.
5. Do function evaluations and update the information.

This sampling framework is shown to have nice convergence properties and has wide adaptability. It can be applied to various metamodel based optimization approaches and search criteria (like the EI, the probability of improvement). Probability distributions such as uniform or normal distributions can be applied to generate candidate points.

Chapter 3

AN ADDITIVE GLOBAL AND LOCAL GAUSSIAN PROCESS MODEL FOR LARGE DATA SETS

3.1 Introduction

In practice, simulation models are widely used to provide an effective and efficient way to evaluate the behavior of real systems. However with the stochastic and complex nature of most real systems, the simulation models can be time consuming to execute. To facilitate the analysis and optimization of these systems, simpler approximations, known as metamodels, that attempt to accurately capture the relationships between the inputs and outputs of the simulation models, are often constructed with a finite number of evaluations. A metamodel is a simplification of a simulation model. The most popular technique used for metamodeling has been based on parametric polynomial response surface approximations. Various other types of metamodels, like multivariate adaptive regression splines, kriging, radial basis functions, artificial neural networks, and Support Vector Regression (SVR), have also been proposed in recent years. Simpson et al. (2001) reviewed the metamodel application in engineering. Li et al. (2010) also provided a

comprehensive comparison of metamodeling approaches that can also be well applied in simulation optimization. Among all these metamodels, the Gaussian Process model, also known as the kriging model (Cressie, 1993), has been increasingly popular in recent years due to its adaptability and efficiency for approximating various highly flexible and nonlinear functional forms, and its unique statistical view of the prediction error, which makes it more useful in simulation optimization (Jones et al., 1998; Kleijnen, 2014). Originated from geo-statistics (Cressie, 1993), the GP model has been widely applied in Design and Analysis of Computer Experiments (DACE) (Sacks et al., 1989; Santner et al., 2003). Beyond the deterministic computer experiments, it has also been widely used in the stochastic simulation through stochastic kriging model (Ankenman et al., 2010) or the modified nugget effect model (Yin et al., 2011).

However, estimating the Gaussian Process model is a computational challenge when the data sets are large. Examples of large data sets can be found in the field of geology, climate or the Internet, where the data sets are updated every day and predictions or decisions have to be made based on all the information available. In simulation, large datasets may be encountered in experiments running on parallel processors, and also in optimization when the number of search points increases quickly as the optimization algorithm progresses. Given the data size of N , estimating the model parameters with traditional methods like the maximum likelihood estimation and estimating the model predictors involves the inversion of a $N \times N$ covariance matrix, which typically requires $O(N^3)$ operations and $O(N^2)$ memory. This becomes computationally intractable for a large N . As such, a desktop computer is unable to handle data sizes larger than several hundreds. Hence, for further application of the Gaussian Process model for a larger size of data, some approximation techniques need to be applied.

In this chapter, we leverage the benefits of a combined approach like the full scale approximation (FSA) (Sang and Huang, 2012) method and partially independent conditional (PIC) method (Snelson and Ghahramani, 2007), and develop a more general combined model that is flexible in modeling systems whose response changes significantly across the design space. The proposed Additive Global and Local Gaussian Process (AGLGP) model incorporates a global GP model and piecewise local GP models into an additive GP model that enables different correlation structures to be captured

3.1. INTRODUCTION

on the global and local levels, and also across different regions of the space. This enables more flexibility in the modeling of systems whose response can dramatically change over the design space. This non-stationarity is common in engineering systems, and can be due to a highly non-linear response surface or the stochastic nature of systems. For example, in production planning, when different models for different components are integrated to generate the finished goods, the system behavior can differ greatly in distinctive design regions. In maritime transportation, one measure of safety for a vessel at sea is the probability of encountering a conflict, and this can change drastically with the angle of turn in heavy traffic regions.

The general idea behind the AGLGP model is to build a global model with a small set of inducing points to capture the global trend and build a local model to capture the residual process from the global model. Our central contribution is to develop a model that is computationally efficient and can capture the nonstationarity with this additive structure. This new model structure not only helps to mitigate some of the computational issues highlighted, but also is highly flexible in capturing nonstationarity across the domain. Different from the works in FSA and PIC, the AGLGP model explicitly models the local residual structures independently and non-identically to enable this flexibility. In this work, we also propose an approach to determine the local regions which has not been previously addressed but assumed given. We further show several nice properties of the additive model, such as identifiability. With its global and local structure and computational efficiency, the AGLGP model can also be well suited for simulation optimization.

This chapter is organized as follows. In the next section, we give the general ideas for our model and proceed to derive in detail the model form and provide a method for the generation of local regions and the estimation of inducing points. We then develop two model estimation schemes. In Section 3.4 we discuss the identifiability of the AGLGP model. Then in Section 3.5, the performance of the AGLGP model is numerically studied and compared with other approximation methods like FSA and PIC.

3.2 Model Formulation

In this section, we first provide the background of the stochastic Gaussian Process model and give an overview of the proposed additive global and local Gaussian Process (AGLGP) model.

3.2.1 Stochastic Gaussian Process Model Basics

The general stochastic Gaussian Process model assumes that the stochastic simulation response can be modeled as a realization of a random process given as

$$y(x) = f(x) + \epsilon(x), \quad (3.1)$$

where $f(x)$ represents the deterministic mean function of the stochastic response (usually measured by the expectation of simulations' performance), and $\epsilon(x)$ is the random noise with mean zero and unknown variance $\epsilon(x) \sim GP(0, \sigma_\epsilon^2(x))$. $f(x)$ can be further decomposed into the process mean function $\mu(x)$ and a spatial process $z(x) \sim GP(0, R(\cdot))$ with variance $R(x_i, x_i) = \sigma^2$ and covariance $R(x_i, x_j) = \sigma^2 \text{corr}(x_i, x_j)$. A popular choice of $\text{corr}(x_i, x_j)$ is *Gaussian correlation function* that assumes $\text{corr}(x_i, x_j, \boldsymbol{\theta}) = \exp(-\boldsymbol{\theta} \|x_i - x_j\|^2)$. Here $\boldsymbol{\theta}$ is the sensitivity parameter of the correlation function. To model the dependence of $\sigma_\epsilon^2(x)$ on location x , it is further assumed that $\epsilon(x)$ and $z(x)$ are independent. $\sigma_\epsilon^2(x)$ can be modeled by a spatial process (Ankenman et al., 2010; Ng and Yin, 2012). Typically, the mean function $f(x)$ is of the experimenter's interest and the predictor for $f(x)$ at any point x_0 can be expressed as

$$\hat{y}(x_0) = \mu(x_0) + \mathbf{r}'(\mathbf{R} + \boldsymbol{\Sigma}_\epsilon)^{-1}(\mathbf{y} - \boldsymbol{\mu}), \quad (3.2)$$

where \mathbf{r} is the covariance of x_0 to all n observed locations, $\mathbf{r} = (R(x_0, x_1), R(x_0, x_2), \dots, R(x_0, x_n))$ and \mathbf{R} represents the covariance between the observed points with the (ij) th item $R(x_i, x_j)$. The observations \mathbf{y} are all n observed sample means and $\boldsymbol{\Sigma}_\epsilon = \text{diag}(\sigma_\epsilon^2(x_1), \dots, \sigma_\epsilon^2(x_n))$. The predictive variance is given by

$$\hat{s}^2(x_0) = E(f(x_0) - \hat{y}(x_0))^2 = \sigma^2 - \mathbf{r}'(\mathbf{R} + \boldsymbol{\Sigma}_\epsilon)^{-1}\mathbf{r}. \quad (3.3)$$

which represents the prediction uncertainty at location x_0 . This uncertainty depends both on the spatial correlation and the noise variance (Ankenman et al., 2010). A similar modified nugget effect model (MNEK) is derived in Yin et al. (2011)

3.2.2 Overview of the AGLGP model

To motivate the ideas of the AGLGP model, we take the example from Xiong et al. (2007), where the function $y(x) = \sin(30(x - 0.9)^4) \cos(2(x - 0.9)) + (x - 0.9)/2$ shown in Figure 3.1 is studied. Here we see that the function has both a large and small scale of dependence, and the mean in the region $x \in [0, 0.4]$ is much smaller than the mean in $x \in [0.4, 1]$. To fit a single GP model with a constant mean throughout the input region, the model will overestimate the mean and underestimate the small scale dependence in $x \in [0, 0.4]$, while underestimate the mean and overestimate the large scale dependence in $x \in [0.4, 1]$.

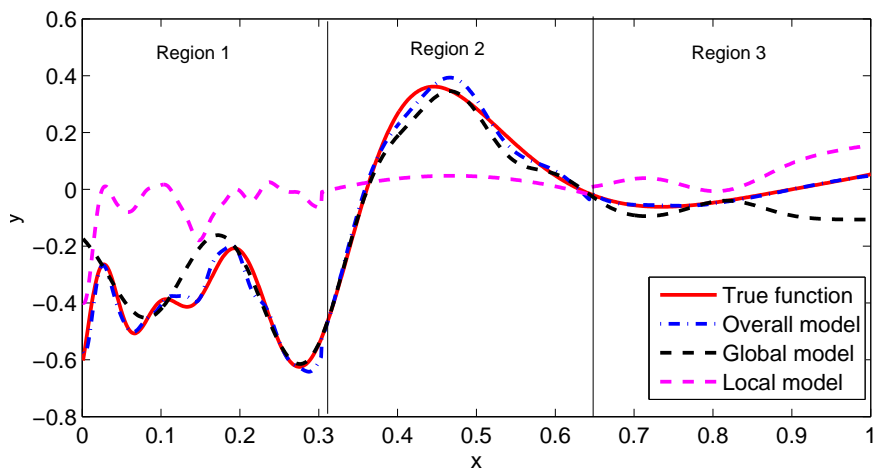


Figure 3.1: Plot of function $y(x) = \sin(30(x - 0.9)^4) \cos(2(x - 0.9)) + (x - 0.9)/2$, the global, local and overall models

To address this, we propose to model the function with a global model, that is developed through a small set of inducing points to capture the long lengthscale global trend and separate local models are to capture the residual process (of shorter lengthscale) in separate local regions (see the dotted fits in Figure 3.4). With a smaller set of inducing points and regional local models, the computational requirements to fit the model are greatly reduced.

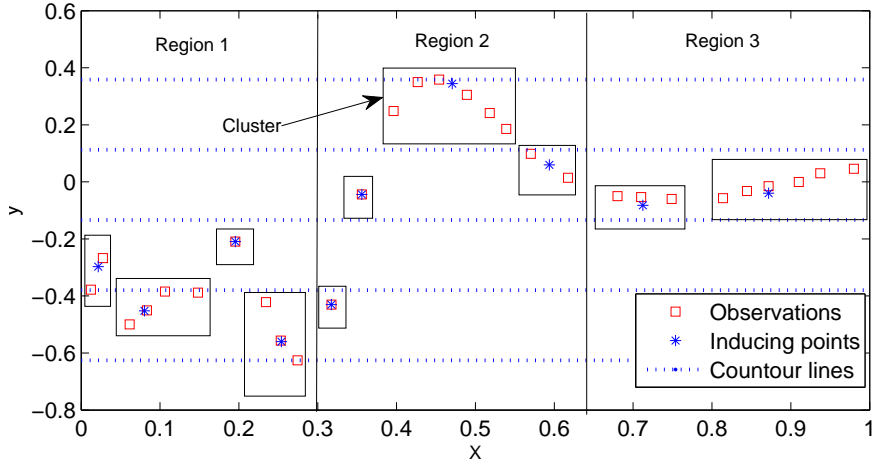


Figure 3.2: The inducing points and local regions

To adequately capture the global trend, the inducing points should sufficiently summarize the observed data points and smooth out the local fluctuations to highlight the global trend. To do so, we will carefully cluster the observed data points into groups based on their location x and observation y and summarize these characteristics in each group with a representative inducing point (usually the centroid of the group). Figure 3.2 illustrates this, where the boxes show the cluster groups and the centroid in each group summarizes the observed data in the cluster. We denote the set of m inducing points by $\mathbf{x}_g = (x_g^1, \dots, x_g^m)$, where x_g^i is a d -dimension vector $x_g^i = (x_g^{i1}, \dots, x_g^{id})$, and denote \mathbf{y}_g as the latent global 'observations' at \mathbf{x}_g . These points are not observed but summarized points of observations.

To capture the local residuals, the idea is to divide the whole design space \mathbb{X}_Ω into K non-overlapping local regions \mathbf{D}_k , $k = 1, \dots, K$, where $\cup_{k=1}^K \mathbf{D}_k = \mathbb{X}_\Omega$. Then the set of evaluations points in each local region \mathbf{D}_k is used to fit a local model in each region. We denote $\mathbf{x}_1^k = (x_l^1, \dots, x_l^{r_k})$ to be the evaluation points in region k , where r_k is the number of evaluation points in region k , and denote the latent local residuals as \mathbf{y}_1^k . Local residuals for all evaluation points are denoted as $\mathbf{y}_1 = (\mathbf{y}_1^1, \dots, \mathbf{y}_1^K)$.

3.2.3 The Development of the AGLGP Model

Based on the stochastic kriging model in Section 3.2.1, we argue that the following model can work as a good approximation of the stochastic kriging model when the data size gets larger. We call it an additive global and local Gaussian Process (AGLGP) model.

3.2. MODEL FORMULATION

The AGLGP models the stochastic simulation response at a point x as

$$y(x) = f(x) + \epsilon(x) = f_{global}(x) + \sum_{k=1}^K w_k f_{local}^k(x) + \epsilon(x) \quad (3.4)$$

$$w_k = \begin{cases} 1, & x \in \mathbf{D}_k \\ 0, & x \notin \mathbf{D}_k \end{cases}$$

Here the process mean of the stochastic response $f(x)$ is decomposed into a global process $f_{global}(x)$, which models the global trend and K local processes, with each local process $f_{local}^k(x)$ modeling the residual process that is unexplained by $f_{global}(x)$ in local region \mathbf{D}_k . $f_{global}(x)$ is assumed to be a stationary GP with mean μ and covariance function $R_g(\cdot)$, while $f_{local}^k(x)$ is assumed to be a stationary GP in local region \mathbf{D}_k with mean 0 and covariance function $R_l^k(\cdot)$, where

$$R_g(x_i, x_j) = \sigma^2 \text{corr}_g(x_i, x_j, \boldsymbol{\theta}), \quad R_l^k(x_i, x_j) = \tau_k^2 \text{corr}_l^k(x_i, x_j, \boldsymbol{\alpha}_k)$$

σ^2 and τ_k^2 are the variances of the global and local processes respectively and $\text{corr}_g(\cdot)$ and $\text{corr}_l^k(\cdot)$ are the correlation structures of the individual processes. Furthermore, $f_{global}(x)$ and $f_{local}^k(x)$ are assumed to be piecewise independent, and different local covariance functions are allowed in different regions. This enables the flexibility to capture the nonstationarity in the mean process. As the global model is expected to capture the long lengthscale global trend and the local model is expected to capture the short lengthscale residual details, it is reasonable to add constraints on the unknown correlation parameters $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ to satisfy $\mathbf{0} \leq \boldsymbol{\theta} \leq \boldsymbol{\alpha}$. This will ensure a smoother global model to capture the global trend.

As the global model $f_{global}(x)$ is a latent process modeling only the inducing points \mathbf{x}_g , it is reasonable to assume a deterministic global model. Given the set of inducing points \mathbf{x}_g and the global evaluations \mathbf{y}_g , the best linear unbiased global predictor at any given point x_0 can then be written as

$$\hat{y}_{global}(x_0) = \mu + \mathbf{g}' \mathbf{G}_m^{-1} (\mathbf{y}_g - \mathbf{1}' \mu), \quad (3.5)$$

where $\mathbf{g} = (R_g(x_0, x_g^1), \dots, R_g(x_0, x_g^m))$, \mathbf{G}_m is $m \times m$ covariance matrix with ij th element $R_g(x_g^i, x_g^j)$. This global predictor interpolates \mathbf{y}_g since $\widehat{y}_{global}(\mathbf{x}_g^j) = \mu + \mathbf{e}_1'(\mathbf{y}_g - \mathbf{1}'\mu) = y_g^j$.

With the fitted global model, we can obtain $\widehat{\mathbf{y}}_{global} = (\widehat{y}_{global}(x_1), \dots, \widehat{y}_{global}(x_n))$. The residuals, which capture both the residuals from the mean function and the random noise, can then be obtained by $\mathbf{y}_1 = \mathbf{y} - \widehat{\mathbf{y}}_{global}$ and modeled by another stochastic kriging model $y_{local} = \sum_{k=1}^K w_k f_{local}^k(x) + \epsilon(x)$. This local model captures the local biases of the global model in each local region and the inherent stochastic noise in the system. As we focus on stochastic simulations, \mathbf{y} is the sample mean of the replications taken at each evaluation point. Given K local regions $\mathbf{D}_1, \dots, \mathbf{D}_K$, the local predictor at any given point x_0 is given by

$$\widehat{y}_{local}(x_0) = \mathbf{l}_k'(\mathbf{L}_k + \mathbf{\Sigma}_\epsilon)^{-1}\mathbf{y}_1^k, \forall x_0 \in \mathbf{D}_k, \quad (3.6)$$

where $\mathbf{l}_k = (R_l^k(x_0, x_l^i), \dots, R_l^k(x_0, x_l^{r_k}))$ and \mathbf{L}_k is the covariance matrix with the (jh) th element $R_l^k(x_l^j, x_l^h), \forall x_l^j, x_l^h \in \mathbf{x}_1^k$. $\mathbf{\Sigma}_\epsilon = \text{diag}(\hat{\sigma}_\epsilon^2(x_l^1), \dots, \hat{\sigma}_\epsilon^2(x_l^{r_k}))$, where $\hat{\sigma}_\epsilon^2(x_l^j)$ can be estimated from the sample variance.

From Equation (3.5) and Equation (3.6), the overall AGLGP predictor can be expressed by

$$\begin{aligned} \widehat{y}(x_0) &= \widehat{y}_{global}(x_0) + \widehat{y}_{local}(x_0) \\ &= \mu + \mathbf{g}'\mathbf{G}_m^{-1}(\mathbf{y}_g - \mathbf{1}'\mu) + \mathbf{l}_k'(\mathbf{L}_k + \mathbf{\Sigma}_\epsilon)^{-1}\mathbf{y}_1^k, \forall x_0 \in \mathbf{D}_k. \end{aligned} \quad (3.7)$$

As \mathbf{y}_g and \mathbf{y}_1^k are latent processes that are not observed directly, this predictor and the predictive distribution of any input x_0 can be estimated by integrating out the random variables \mathbf{y}_g and \mathbf{y}_1^k . The details will be described in Section 3.3.

3.2.4 Selection of Inducing Points and Local Regions

In order to develop the AGLGP model in Equation (3.7), two important initial steps have to be taken. Firstly, a smaller set of inducing points have to be determined from the large set of evaluation points to fit the global model. Secondly, the whole space has

3.2. MODEL FORMULATION

to be divided into local regions for the local models. This will facilitate the estimation of the model and enable better modeling of nonstationary responses. Two desirable characteristics of these steps are:

- The inducing points should be generated to represent similar evaluation points around them. The idea is to have these points sufficiently represent neighboring observations around them and enable the capture of the long lengthscale global trend across regions.
- Local regions are divided to provide the largest separation between observed evaluation points. This can better approximate the assumption of independence of the local processes across regions.

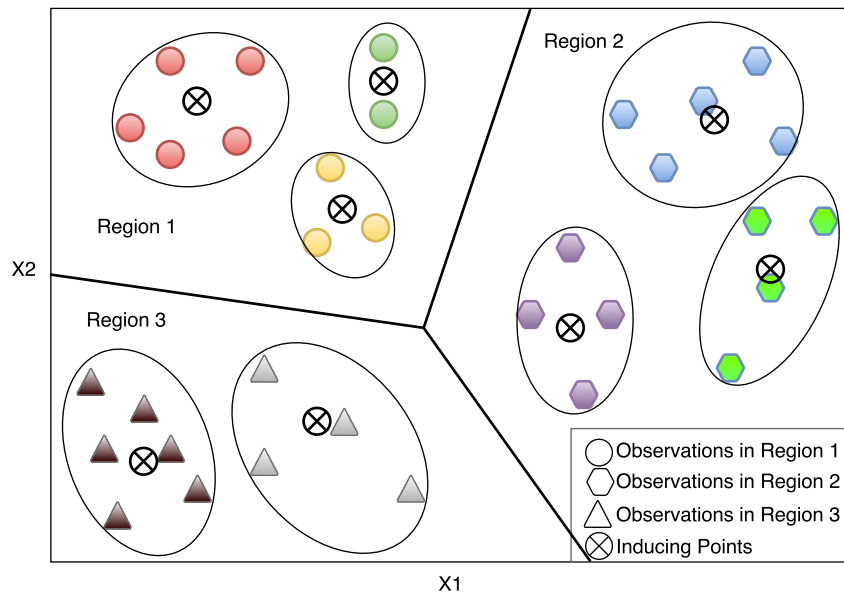


Figure 3.3: Inducing points and local regions

Figure 3.3 illustrates how the data points can be divided into smaller clusters that are represented by an inducing point each, and how the whole two dimensional space is divided into 3 local regions. In the following subsections, we present one particular selection procedure where the observations are grouped through k -means, the regions are classified by Support Vector Machine (SVM) based on the groups, and each group of points are further clustered based on a set of contour lines to generate inducing points. Other reasonable selection criteria such as Voronoi tessellation can also be applied.

3.2.4.1 Partitioning the Design Space into Local Regions

Here we illustrate an approach to divide the whole design space \mathbb{X}_Ω into K different local regions $\{\mathbf{D}_k\}_{k=1,\dots,K}$ that provides the largest separation between observations. Firstly evaluation points \mathbf{x} are clustered through K -means such that each cluster comprises of a group of points \mathbf{x}_l^k whose inter-point *Euclidean* distances are small compared with the distances to points outside the cluster. Hence, each point in the design space \mathbb{X}_Ω is assigned to one and only one of the discrete K clusters. The design space is thereafter divided into K local regions \mathbf{D}_k . To do so, we construct the boundaries through classification of the clusters with Support Vector Machine (SVM), which chooses the best hyperplane that represents the largest separation or margin between two neighboring clusters. For multiple local regions, we generate pairwise classifiers. Suppose that two hyperplanes that separate two clusters of data sets \mathbf{x}_l^i and $\mathbf{x}_l^j, i \neq j$ with no points between them, are described by a set of points \mathbf{x}_{p_1} and \mathbf{x}_{p_2} that satisfies $\mathbf{w} \cdot \mathbf{x}_{p_2} - b = 1$ and $\mathbf{w} \cdot \mathbf{x}_{p_1} - b = -1$, where \mathbf{w} is the normal vector to the hyperplane. Parameters are then optimized by maximizing the distance between these two hyperplanes $\frac{2}{\|\mathbf{w}\|}$, which is equivalent to minimizing $\|\mathbf{w}\|$ given the constraints $\mathbf{w} \cdot x - b \geq 1, x \in \mathbf{x}_l^i$ and $\mathbf{w} \cdot x - b \leq -1, x \in \mathbf{x}_l^j$ to ensure no data between the two hyperplanes. This approach will create regions that have the largest separation, which achieves the desirable characteristic for local regions. The k -means clustering followed by the SVM separation can be executed quite efficiently with R or MATLAB.

3.2.4.2 Determining Inducing Points

After the evaluation points have been clustered into different local regions, we further require a small number of inducing points to represent the points in each local region. The set of points \mathbf{x}_l^k in each local region \mathbf{D}_k are further divided into clusters based on their observation values \mathbf{y}^k . To do this we first determine the range of observations in the whole space $\Delta y = \max(\mathbf{y}) - \min(\mathbf{y})$ and a user-defined range Δ within each cluster. To make sure each cluster has a similar range of observations, Δ is selected such that $\Delta y / \Delta$ takes an integer. Then contour lines are drawn with an interval of Δ , i.e. $L = \{y | y = c\}$ where $c \in \{\min(\mathbf{y}), \min(\mathbf{y}) + \Delta, \dots, \max(\mathbf{y}) - \Delta, \max(\mathbf{y})\}$. Finally, the set of points

settled between two neighboring contour lines $\mathbf{c}_j^k = \{x | \min(\mathbf{y}) + \Delta \times (j - 1) \leq y(x) \leq \min(\mathbf{y}) + \Delta \times j\}$, $j = 1, \dots, \Delta y / \Delta$ are further grouped into smaller clusters. If $\exists x_l \notin \mathbf{c}_j^k$ such that $\|x_h - x_k\| > \|x_l - x_h\|$, $\forall x_h, x_k \in \mathbf{c}_j^k$, $x_h \neq x_k$, \mathbf{c}_j^k will be further divided into two subclusters between x_h and x_k . With this, we will end up with m subclusters $\mathbf{c} = (c_1, \dots, c_m)$. The average of all the evaluation points in each subclusters will be the inducing point value. This approach will generate inducing points that are representative of the evaluation points in each subcluster because they are similar in both the x and y space.

In order to determine the regions and inducing points with the above schemes, two key inputs need to be specified, specifically the number of inducing points and the number of regions. Here, we recommend the number of inducing points selected to be under a hundred (for computational efficiency), and the number of local regions to be within 10. This however has to be traded-off with the data size and the number of parameters to be estimated as the number of parameters increases with the number of local regions.

3.3 Model Estimation

In order to apply the AGLGP model for simulation metamodeling, a method to estimate the model parameters $\phi = [\mu, \sigma^2, \theta, \tau^2, \alpha]$ is required. Here, we apply the most commonly used approach, the maximum likelihood method. The conditional likelihood function of the observations \mathbf{y} given $\mathbf{x}_g, \mathbf{y}_g, \mathbf{y}_l$ can be derived as

$$L(\phi) = \frac{1}{(2\pi)^{n/2} |\mathbf{R}|^{1/2}} \exp[(\mathbf{y} - \hat{\mathbf{y}})' \mathbf{R}^{-1} (\mathbf{y} - \hat{\mathbf{y}})], \quad (3.8)$$

where $\mathbf{R} = \mathbf{\Lambda} + \mathbf{\Gamma} + \mathbf{\Sigma}_\epsilon$ and $\hat{\mathbf{y}} = \boldsymbol{\mu} + \mathbf{G}_{nm} \mathbf{G}_m^{-1} (\mathbf{y}_g - \boldsymbol{\mu}) + \mathbf{L}'_n (\mathbf{L}_n + \mathbf{\Sigma}_\epsilon)^{-1} \mathbf{y}_l$. \mathbf{y}_g are global observations at inducing points \mathbf{x}_g and \mathbf{y}_l are the local observations at all observed points \mathbf{x} . $\mathbf{\Lambda}$ and $\mathbf{\Gamma}$ represent the mean square prediction error for the global model and local models respectively, with $\mathbf{\Lambda} = \mathbf{G}_{nn} - \mathbf{G}'_{nm} \mathbf{G}_m^{-1} \mathbf{G}_{mn}$, $\mathbf{\Gamma} = \mathbf{L}_n - \mathbf{L}_n (\mathbf{L}_n + \mathbf{\Sigma}_\epsilon)^{-1} \mathbf{L}_n$. However, as \mathbf{y}_g and \mathbf{y}_l are latent variables that are not observed, maximizing Equation (3.8) is not possible. Instead, the marginal likelihood of \mathbf{y} is used.

We restate here the structure of the AGLGP model, where $f_{global}(x)$ is modeled as a stationary GP with mean μ and covariance function $R_g(\cdot)$, while $f_{local}^k(x)$ is modeled as a stationary GP in each local region \mathbf{D}_k with mean 0 and covariance function $R_l^k(\cdot)$. Hence the latent variables \mathbf{y}_g and \mathbf{y}_l satisfy the following distributions

$$\mathbf{y}_g | \mathbf{x}_g \sim N(\boldsymbol{\mu}, \mathbf{G}_m), \quad \mathbf{y}_l | \mathbf{x} \sim N(\mathbf{0}, \mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon). \quad (3.9)$$

where $\mathbf{L}_n = \text{diag}\{\mathbf{L}_1, \dots, \mathbf{L}_K\}$ is block diagonal matrix with each block \mathbf{L}_k representing the local correlations within a local region \mathbf{D}_k . \mathbf{G}_m and \mathbf{L}_n are functions of global inducing points \mathbf{x}_g and the local regions.

In this section, we describe two estimation methods and derive their unconditional predictive distribution at any new point x_0 . The first method considers both the global and local models in a single stage that accounts for interactions between both the models, and balances their effect. The second is a much faster two-stage estimation, which estimates the global parameters and local parameters separately.

3.3.1 Estimating Predictive Distribution and Parameters in a One Stage Approach

3.3.1.1 Estimating Predictive Distribution in One Stage

The unconditional predictive distribution of any input x_0 can be obtained by integrating out the latent variables \mathbf{y}_g and \mathbf{y}_l . We first observe from Equation (3.7) that the conditional predictive distribution of the simulation response y_0 at any evaluation point x_0 is dependent on \mathbf{y}_g and \mathbf{y}_l , and is given as

$$y_0 | x_0, \mathbf{x}_g, \mathbf{y}_g, \mathbf{x}, \mathbf{y}_l \sim N(\mu + \mathbf{g}' \mathbf{G}_m^{-1} (\mathbf{y}_g - \boldsymbol{\mu}) + \mathbf{l}' (\mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon)^{-1} \mathbf{y}_l, \lambda + \gamma + \sigma_\epsilon^2(x_0)), \quad (3.10)$$

where the mean square prediction error $\lambda = G_{nn} - \mathbf{g}' \mathbf{G}_m^{-1} \mathbf{g}$, $\gamma = L_{nn} - \mathbf{l}' \mathbf{L}_n^{-1} \mathbf{l}$, and G_{nn} and L_{nn} are the global and local model variances at location x_0 .

Next we note that the conditional distribution of \mathbf{y}_g given \mathbf{x}, \mathbf{y} can be shown to be

$$\mathbf{y}_g | \mathbf{x}_g, \mathbf{x}, \mathbf{y} \sim N(\boldsymbol{\mu} + \mathbf{G}_m \mathbf{Q}_m^{-1} \mathbf{G}_{mn} \mathbf{K}^{-1} (\mathbf{y} - \boldsymbol{\mu}), \mathbf{G}_m \mathbf{Q}_m^{-1} \mathbf{G}_m), \quad (3.11)$$

where $\mathbf{Q}_m = \mathbf{G}_m + \mathbf{G}_{mn} \mathbf{K}^{-1} \mathbf{G}_{nm}$ and $\mathbf{K} = \mathbf{L}_n + \boldsymbol{\Lambda} + \boldsymbol{\Sigma}_\epsilon$, $\boldsymbol{\Lambda} = \text{diag}\{\mathbf{G}_n - \mathbf{G}_{nm} \mathbf{G}_m^{-1} \mathbf{G}_{mn}\}$, and the conditional distribution of y_1 given \mathbf{y}_g and \mathbf{y} is given as

$$\begin{aligned} y_1 | \mathbf{x}_g, \mathbf{y}_g, \mathbf{x}, \mathbf{y} \\ \sim N(\mathbf{L}_n \mathbf{K}^{-1} \{\mathbf{y} - \boldsymbol{\mu} - \mathbf{G}_{nm} \mathbf{G}_m^{-1} (\mathbf{y}_g - \boldsymbol{\mu})\}, \mathbf{L}_n - \mathbf{L}_n \mathbf{K}^{-1} \mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon), \end{aligned} \quad (3.12)$$

Then given a new input x_0 , the unconditional predictive distribution can be obtained by integrating $\mathbf{y}_g, \mathbf{y}_1$ from Equation (3.10). This gives $y_0 | x_0, \mathbf{x}_g, \mathbf{x}, \mathbf{y} \sim N(\hat{y}(x_0), \hat{s}^2(x_0))$, where the predictive mean $\hat{y}(x_0)$ and predictive variance $\hat{s}^2(x_0)$ are

$$\begin{aligned} \hat{y}(x_0) = & \boldsymbol{\mu} + [\mathbf{g}' \mathbf{Q}_m^{-1} \mathbf{G}_{mn} + \mathbf{l}' (\mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon)^{-1} \mathbf{L}_n \mathbf{K}^{-1} (\mathbf{K} - \mathbf{G}_{nm} \mathbf{Q}_m^{-1} \mathbf{G}_{mn})] \\ & \times \mathbf{K}^{-1} (\mathbf{y} - \boldsymbol{\mu}), \end{aligned} \quad (3.13)$$

$$\begin{aligned} \hat{s}^2(x_0) = & G_{nn} - \mathbf{g}' (\mathbf{G}_m^{-1} - \mathbf{Q}_m^{-1}) \mathbf{g} + \frac{(\mathbf{1} - \mathbf{l}' \mathbf{G}_m^{-1} \mathbf{g})^2}{\mathbf{l}' \mathbf{G}_m^{-1} \mathbf{1}} \\ & + L_{nn} - \mathbf{l}' [\boldsymbol{\Sigma}_\epsilon + \mathbf{L}_n]^{-1} \mathbf{L}_n \mathbf{K}^{-1} \mathbf{L}_n [\boldsymbol{\Sigma}_\epsilon + \mathbf{L}_n]^{-1} \mathbf{1} \end{aligned} \quad (3.14)$$

The detailed derivation is shown in Appendix A. In the heterogeneous case where the random error is assumed to be independent but not identical, variance information is not available to estimate $\sigma_\epsilon^2(x_0)$ unless the location has been previously observed. Here we propose to model $\log(\sigma_\epsilon^2)$ as a Gaussian Process. This natural log transformation has the nice properties of approximating normality, stabilizing variance, and ensuring inverse transformation back to the positive scale. The details can be found in Ng and Yin (2012).

Observing the covariance structure in more detail, suppose the full covariance between observations \mathbf{y} is assumed to be \mathbf{G}_n . From Equation (3.13), the AGLGP model approximates this with $\mathbf{G}_{nm} \mathbf{G}_m^{-1} \mathbf{G}_{mn} + \text{diag}\{\mathbf{G}_n - \mathbf{G}_{nm} \mathbf{G}_m^{-1} \mathbf{G}_{mn}\} + \mathbf{L}_n$. We see that the global reduced rank approximation $\mathbf{G}_{nm} \mathbf{G}_m^{-1} \mathbf{G}_{mn}$ is only able to capture the long lengthscale correlations, and can greatly underestimate the covariance

\mathbf{G}_n when the number of inducing points is small. The last two terms on the other hand can be viewed as local adjustments to the reduced rank approximation. The term $\text{diag}\{\mathbf{G}_n - \mathbf{G}_{nm}\mathbf{G}_m^{-1}\mathbf{G}_{mn}\}$ adjusts mainly the diagonal terms for the over or underestimation of the variance and \mathbf{L}_n captures the correlation between points in the same local region. Different from the PIC model, whose global approximation and local adjustment share the same set of parameters and the parameter estimation might offset the effect of these two components, the AGLGP model is more robust in estimation by enabling different residual correlation structures in the different local regions.

Next, we list some results and observations from two special cases.

Firstly, suppose the parameters $\boldsymbol{\theta}, \boldsymbol{\alpha}, \sigma^2, \boldsymbol{\tau}^2$ are known. We can obtain the following theorem.

Theorem 3.1. *Given the parameter values $\boldsymbol{\theta}, \boldsymbol{\alpha}, \sigma^2, \boldsymbol{\tau}^2$, the predictive mean $\hat{y}(x_0)$ is an unbiased predictor of the process mean, i.e. $E[\hat{y}(x_0) - y(x_0)] = 0$.*

The proof of this theorem is provided in the Appendix B.

Next consider a second case where $m = n$, $\mathbf{x}_g = \mathbf{x}$ and $k = 1$. Here we can show the following proposition.

Proposition 3.2. *When $m = n$, $\mathbf{x}_g = \mathbf{x}$ and $k = 1$, the global model sufficiently captures the whole process mean $y_{\text{global}}(x) = f(x)$, and the local model includes only the random noise $y_{\text{local}}(x) = \epsilon(x)$. Then, the predictive mean (3.13) will reduce to the Stochastic Kriging and MNEK predictor.*

The proof can be found in the Appendix C.

3.3.1.2 Estimating Parameters in One Stage

The predictive distributions derived in Equation (3.10) are given under the assumption that the parameters are known. To derive the maximum-likelihood estimator of $\boldsymbol{\theta}, \boldsymbol{\alpha}, \sigma^2, \boldsymbol{\tau}^2$, we first derive the marginal likelihood of \mathbf{y} by integrating out \mathbf{y}_g from Equation (3.8) to get

$$\mathbf{y}|\mathbf{x}, \mathbf{x}_g \sim N(\boldsymbol{\mu}, \mathbf{G}'_{mn}\mathbf{G}_m^{-1}\mathbf{G}_{mn} + \boldsymbol{\Lambda} + \mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon) \quad (3.15)$$

Then the negative log-likelihood function which is dependent on $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ can be shown to be

$$l(\boldsymbol{\theta}, \boldsymbol{\alpha}, \sigma^2, \boldsymbol{\tau}^2) = \frac{1}{2} \ln \det \mathbf{R} + \frac{1}{2} (\mathbf{y} - \hat{\boldsymbol{\mu}})' \mathbf{R}^{-1} (\mathbf{y} - \hat{\boldsymbol{\mu}}) \quad (3.16)$$

where $\mathbf{R} = \mathbf{G}'_{mn} \mathbf{G}_m^{-1} \mathbf{G}_{mn} + \boldsymbol{\Lambda} + \mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon$. The evaluation of this log-likelihood requires calculation of the inverse and determinant of the $n \times n$ matrix \mathbf{R} . From the Woodbury formula, we get

$$\begin{aligned} & [\mathbf{G}_{nm} \mathbf{G}_m^{-1} \mathbf{G}_{mn} + \boldsymbol{\Lambda} + \mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon]^{-1} \\ &= (\mathbf{I} - [\boldsymbol{\Lambda} + \mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon]^{-1} \mathbf{G}_{nm} \mathbf{Q}_m^{-1} \mathbf{G}_{mn}) [\boldsymbol{\Lambda} + \mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon]^{-1} \end{aligned} \quad (3.17)$$

where \mathbf{L}_n is a block diagonal matrix (each block representing a local region) and $\boldsymbol{\Lambda} + \boldsymbol{\Sigma}_\epsilon$ is a diagonal matrix, so $\mathbf{L}_n + \boldsymbol{\Lambda} + \boldsymbol{\Sigma}_\epsilon$ can be inverted in blocks. Hence, the right-hand side of Equation (3.17) involves only inversion and multiplication with a sparse $n \times n$ block diagonal matrix $\boldsymbol{\Lambda} + \mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon$ as well as the inversion of a $m \times m$ matrix \mathbf{Q}_m . Although there is no requirement for the blocks to be of equal size, for illustration, suppose they all have size B . Thus the computational complexity of the log-likelihood calculation is of the order $O(nm^2 + nB^2)$. By using a small number m , the computational cost in fitting the spatial model can be greatly reduced relative to the expensive computational cost of using the original covariance function \mathbf{G}_n , where the computational complexity is typically of the order $O(n^3)$. Besides, the inverse of $\mathbf{L}_n + \boldsymbol{\Lambda} + \boldsymbol{\Sigma}_\epsilon$ shows higher numerical stability compared with $\mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon$, especially when the noise variance is small. In addition, the covariance matrix can easily become ill-conditioned, for example, when the design points are close to each other. In this case, the popular approach is to add a nonzero nugget to the diagonal elements of the covariance matrix. $\boldsymbol{\Lambda}$ naturally serves this purpose in the AGLGP model, making it numerically more stable.

To minimize the negative log-likelihood function in Equation (3.16), optimization algorithms like the quasi-Newton methods can be applied. To address the issue of the increasing number of parameters to estimate when input dimension increases, a further assumption such as $\alpha_{ih} = \theta_i + \kappa_h$ (assuming a constant difference κ_h from θ_i) can be made (Ba and Joseph, 2012). We further suggest to start the algorithm from multiple starting points to improve the convergence of these optimization methods.

It is also noted that as the number of parameters to be optimized increases with the number of local regions, the approximations of the Hessian matrix can become a computational burden. However, based on our assumptions of the independence across local regions, most of the terms in the Hessian matrix are zeros. To illustrate this, suppose we have K local regions and each design has a dimension of d . Compared to the original GP model whose likelihood contains only $d + 1$ unknown parameters, the likelihood of AGLGP model contains $(K + 1)(d + 1)$ unknown parameters. The Hessian matrix of this would be a $(K + 1)(d + 1) \times (K + 1)(d + 1)$ matrix. From the corresponding independence assumption across local regions, the Hessian terms $\frac{\partial^2 l}{\partial \alpha_{ih} \partial \alpha_{jk}} = 0, \forall h \neq k$, where α_{ih}, α_{jk} represent the local parameters of input dimensions i and j in local regions h and k . Hence, there are $(K^2 - K)(d + 1)^2$ items in this Hessian matrix that are zeros, improving the computation.

Although the global model is continuous and smooth throughout the input space, the overall AGLGP model is still discontinuous at local region boundaries. The discontinuities however are much smaller than with local models alone. These small discontinuities are usually not significant for the purpose of predictions. However, if the continuity in the overall model is required, further continuity restrictions might be required in the parameter estimation. For example, Park et al. (2011) smoothed the discontinuities in local models by adding extra constraints on the subregion boundaries when combining the local predictors. However substantial computational time is required to determine the value at the boundaries. Continuity can also be achieved by releasing the independence assumption over non-overlapping local regions. The original matrix can instead be partitioned into overlapping subregions and this smooths out the subregions. However, the inverse of the new local matrix will become more computationally complex, and the larger the overlapping areas, the more complicated the matrix inversion. Another alternative is to constrain the local model predictors to be zero at the boundaries, and have only the global model to dominate at the boundaries.

3.3.2 Estimating the Predictive Distribution and Parameters with a Faster Two Stage Approach

When estimating all the parameters in one stage through deriving the marginal likelihood, the maximization of likelihood may be impractical due to the number of parameters. To further improve on the computation, an alternative two stage estimation approach first estimates the global trend model $\hat{y}_g(x_0)$ and its parameters from the marginal likelihood of the global model (after integrating out only the latent variable \mathbf{y}_g). The predicted residuals $\mathbf{y}_1 = \mathbf{y} - \hat{\mathbf{y}}_g$ for the local model are then obtained from the predictions of the global model and the observations \mathbf{y} . The parameters of the local model are then obtained from the likelihood of the local model with the predicted residuals. This achieves optimization of the models separately, providing only local optimal parameters for the overall model.

First we consider the likelihood of the global model only. After we integrating out \mathbf{y}_g , it reduces to the Sparse Gaussian Process model (Snelson and Ghahramani, 2005) and we have the marginal likelihood as

$$\mathbf{y}|\mathbf{x}_g \sim N(\boldsymbol{\mu}, \mathbf{G}_{nm}\mathbf{G}_m^{-1}\mathbf{G}_{mn} + \boldsymbol{\Lambda} + \boldsymbol{\Sigma}_\epsilon) \quad (3.18)$$

and the mean and variance of the predictive distribution at any point x_0 are given as

$$\hat{y}_g(x_0) = \boldsymbol{\mu} + \mathbf{g}'\mathbf{Q}_m^{-1}\mathbf{G}_{mn}(\boldsymbol{\Lambda} + \boldsymbol{\Sigma}_\epsilon)^{-1}(\mathbf{y} - \mathbf{1}'\boldsymbol{\mu}) \quad (3.19)$$

$$\hat{s}_g^2(x_0) = [G_{nn} - \mathbf{g}'\mathbf{G}_m^{-1}\mathbf{g}] + \mathbf{g}'\mathbf{Q}_m^{-1}\mathbf{g} \quad (3.20)$$

where $\mathbf{Q}_m = \mathbf{G}_m + \mathbf{G}_{mn}(\boldsymbol{\Lambda} + \boldsymbol{\Sigma}_\epsilon)^{-1}\mathbf{G}_{nm}$. Hence, as the deterministic model is assumed, the global model interpolates at the inducing points to give the global trend. However, since \mathbf{y}_g are latent variables, by integrating out \mathbf{y}_g , the global predictors can be estimated through noisy observations \mathbf{y} (Equation (3.19)). This noise is reflected in the mean squared prediction error in Equation (3.20). The term in brackets in Equation (3.20) is mean square error of the deterministic GP model, which takes a value of zero at inducing points, but the second term in the right hand-side of Equation (3.20) is positive, reflecting how the intrinsic noise inflates the mean squared error.

The local models are then estimated to get the predictive local residuals. The local residuals satisfy $\mathbf{y}_1 \sim N(\mathbf{y} - \hat{\mathbf{y}}_g, \hat{\Sigma}_g^2)$, where $\mathbf{y} - \hat{\mathbf{y}}_g = \mathbf{y} - \mathbf{1}'\mu - \mathbf{G}_{nm}\mathbf{Q}_m^{-1}\mathbf{G}_{mn}(\mathbf{\Lambda} + \mathbf{\Sigma}_\epsilon)^{-1}(\mathbf{y} - \mathbf{1}'\mu)$. We fit a local GP model given \mathbf{y}_1 and get a conditional distribution of $\hat{y}_l(x_0)$ at any given point x_0 as

$$\hat{y}_l(x_0)|\mathbf{y}_1 \sim N(\mathbf{l}'(\mathbf{L}_n + \mathbf{\Sigma}_\epsilon)^{-1}\mathbf{y}_1, \mathbf{l}'(\mathbf{L}_n + \mathbf{\Sigma}_\epsilon)^{-1}\mathbf{1}). \quad (3.21)$$

The local predictive mean and the local predictive variance at any point x_0 is derived as

$$\begin{aligned} \hat{y}_l(x_0) &= \mathbf{l}'(\mathbf{L}_n + \mathbf{\Sigma}_\epsilon)^{-1}(\mathbf{\Lambda} + \mathbf{\Sigma}_\epsilon - \mathbf{G}_{nm}\mathbf{Q}_m^{-1}\mathbf{G}_{mn})(\mathbf{\Lambda} + \mathbf{\Sigma}_\epsilon)^{-1}(\mathbf{y} - \mathbf{1}'\mu) \\ \hat{s}_l^2(x_0) &= \mathbf{l}'(\mathbf{L}_n + \mathbf{\Sigma}_\epsilon)^{-1}\mathbf{1} \end{aligned}$$

The two stage predictive mean $\hat{y}(x_0) = \hat{y}_g(x_0) + \hat{y}_l(x_0)$ and predictive variance $\hat{s}^2(x_0) = \hat{s}_g^2(x_0) + \hat{s}_l^2(x_0)$ are then derived as

$$\begin{aligned} \hat{y}(x_0) &= \mu + [\mathbf{g}'\mathbf{Q}_m^{-1}\mathbf{G}_{mn} + \mathbf{l}'(\mathbf{L}_n + \mathbf{\Sigma}_\epsilon)^{-1}(\mathbf{\Lambda} + \mathbf{\Sigma}_\epsilon - \mathbf{G}_{nm}\mathbf{Q}_m^{-1}\mathbf{G}_{mn})] \\ &\quad \times (\mathbf{\Lambda} + \mathbf{\Sigma}_\epsilon)^{-1}(\mathbf{y} - \mathbf{1}'\mu), \end{aligned} \quad (3.22)$$

$$\hat{s}^2(x_0) = [G_{nn} - \mathbf{g}'\mathbf{G}_m^{-1}\mathbf{g}] + \mathbf{g}'\mathbf{Q}_m^{-1}\mathbf{g} + \mathbf{l}'(\mathbf{L}_n + \mathbf{\Sigma}_\epsilon)^{-1}\mathbf{1} \quad (3.23)$$

where $\mathbf{Q}_m = \mathbf{G}_m + \mathbf{G}_{mn}(\mathbf{\Lambda} + \mathbf{\Sigma}_\epsilon)^{-1}\mathbf{G}_{nm}$. For the two stage estimation, we can also express the global predictive mean by $\hat{y}_g(x_0) = \hat{\mu} + \mathbf{g}'\mathbf{G}_m^{-1}\mathbf{G}_{mn}\mathbf{R}^{-1}(\mathbf{y} - \mathbf{1}'\hat{\mu})$, where $\mathbf{R} = \mathbf{G}_{nm}\mathbf{G}_m^{-1}\mathbf{G}_{mn} + \mathbf{\Lambda} + \mathbf{\Sigma}_\epsilon$, and the local predictive mean can be derived as $\hat{y}_l(x_0) = \mathbf{l}'(\mathbf{L}_n + \mathbf{\Sigma}_\epsilon)^{-1}\mathbf{y}_1$. Similarly, we can express the global predictive mean as a linear combination of \mathbf{y} , where $\hat{y}_g(x_0) = \sum_{i=1}^n \lambda_i y_i$, $\sum_{i=1}^n \lambda_i = 1$. So $E[\hat{y}_g(x_0)] = E[\sum_{i=1}^n \lambda_i y_i] = \sum_{i=1}^n \lambda_i E[y_i] = \mu$. The local predictive mean, which is a linear combination of \mathbf{y}_1 , can be expressed as $\hat{y}_l(x_0) = \sum_{i=1}^n \delta_i y_l^i$, $\sum_{i=1}^n \delta_i = 1$. So $E[\hat{y}_l(x_0)] = E[\sum_{i=1}^n \delta_i y_l^i] = \sum_{i=1}^n \delta_i E[y_l^i] = 0$. Thus the overall predictive mean has $E(\hat{y}(x_0)) = E(\hat{y}_g(x_0)) + E(\hat{y}_l(x_0)) = E(y(x_0))$, indicating that the two stage predictor is still an unbiased predictor.

In the two stage estimation procedure, we also derive maximum-likelihood estimators (MLEs) for the unknown parameters. But differently, $\mu, \boldsymbol{\theta}, \sigma^2$ are estimated by minimizing the negative log-likelihood function of observations while local parameters $\tau^2, \boldsymbol{\alpha}$ are estimated by minimizing the negative log-likelihood function of local residuals. Specifically, the negative log-likelihood function of $\mu, \boldsymbol{\theta}, \sigma^2$ based on the global model can be written as

$$l(\mu, \boldsymbol{\theta}, \sigma^2) = \frac{1}{2} \ln \det[\mathbf{G}_{nm} \mathbf{G}_m^{-1} \mathbf{G}_{mn} + \boldsymbol{\Lambda} + \boldsymbol{\Sigma}_\epsilon] + \frac{1}{2} (\mathbf{y} - \mathbf{1}'\mu)' [\mathbf{G}_{nm} \mathbf{G}_m^{-1} \mathbf{G}_{mn} + \boldsymbol{\Lambda} + \boldsymbol{\Sigma}_\epsilon]^{-1} (\mathbf{y} - \mathbf{1}'\mu)$$

The global parameters $\mu, \boldsymbol{\theta}, \sigma^2$ are then derived by minimizing this negative log-likelihood function. Then, the local residuals are calculated to derive the negative log-likelihood function of $\tau^2, \boldsymbol{\alpha}$, where

$$l(\tau^2, \boldsymbol{\alpha}) = \frac{1}{2} \ln \det[\mathbf{L} + \boldsymbol{\Sigma}_\epsilon] + \frac{1}{2} \mathbf{y}_1' [\mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon]^{-1} \mathbf{y}_1$$

and calculating the log-likelihood function of local residuals requires an order of $O(nB^2)$. So calculating the log-likelihood function in two stage requires the same computational complexity $O(nm^2 + nB^2)$ as in one stage.

Overall, this two stage approach is computationally faster than the one stage approach as it optimizes the global and local parameters independently. The one-stage estimation approach takes into account the interactions between global model and local model, and there are $(3K + 1)(d + 1)^2$ items in the Hessian matrix that needs to be calculated. In the two stage estimation approach, only a $(d + 1) \times (d + 1)$ Hessian matrix of the second-order derivative of the log-likelihood function needs to be calculated for the global model, and the local model estimation requires additional calculation of $K(d + 1) \times (d + 1)$ Hessian matrices. Hence in the iterative maximization of likelihood function, the two stage estimation will require less than half the effort to calculate its Hessian matrix in each iteration. This two stage estimation approach however is unable to capture the interactions between the global and local models in the optimization of the likelihood.

In general, the one stage estimation approach will provide more accurate predictions as it fully accounts for the interactions between the global and local models in its estimation, and hence, a trade-off between accuracy and computational efficiency has to be considered when selecting the estimation method.

3.4 Identifiability of the AGLGP model

An important property for an additive model to have is identifiability, especially when each component of the model is to be applied for purposes like optimization. To show this property of our model, we assume every possible structure of the AGLGP model is described by parameters $\phi = [\boldsymbol{\mu}, \sigma^2, \boldsymbol{\theta}, \boldsymbol{\tau}^2, \boldsymbol{\alpha}]$. For each ϕ , any set of observations \mathbf{y} follow the multivariate normal distribution $\mathbf{y} \sim N(\boldsymbol{\mu}, \mathbf{R}(\phi))$, so the probability density function for \mathbf{y} is described by $f(\mathbf{y}, \phi)$. Based on Koopmans and Reiersol (1950), we define the identifiability in our framework as follows.

Definition 3.3. Two sets of parameters (structures) ϕ^1 and ϕ^2 are said to be observationally equivalent if $f(\mathbf{y}, \phi^1) = f(\mathbf{y}, \phi^2)$ for all \mathbf{y} in \mathbf{R}^n .

Definition 3.4. Two sets of parameters (structures) ϕ^1 and ϕ^2 for the AGLGP model are said to be observationally equivalent if $\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2$ and $\mathbf{R}(\phi^1) = \mathbf{R}(\phi^2)$ for all \mathbf{y} .

Definition 3.5. A set of parameters ϕ^0 in Φ is said to be globally identifiable if there is no other ϕ in Φ which is observationally equivalent.

The covariance structures for the global and local models are given as $R_g(x_i, x_j) = \sigma^2 \text{corr}_g(|x_i - x_j|, \boldsymbol{\theta})$ and $R_l^k(x_i, x_j) = \tau_k^2 \text{corr}_k^l(|x_i - x_j|, \boldsymbol{\alpha}_k)$. Here we make the following assumptions.

Assumption 3.6. $\text{corr}_g(\cdot), \text{corr}_k^l(\cdot)$ are not linear functions of $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}_k$

Assumption 3.7. The mean $\boldsymbol{\mu}$ is a function that is identifiable and independent of $[\sigma_2^2, \boldsymbol{\theta}_2, \boldsymbol{\tau}_2^2, \boldsymbol{\alpha}_2]$

To show the AGLGP model is identifiable, we need to show $\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2$ and $\mathbf{R}(\phi^1) = \mathbf{R}(\phi^2)$ are only satisfied when $\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2$ and $\phi^1 = \phi^2$. Given the specified global and local covariance structures, we see that \mathbf{R} is not a function of $\boldsymbol{\mu}$. From Assumption 3.7,

3.4. IDENTIFIABILITY OF THE AGLGP MODEL

this is equivalent to showing $\mathbf{R}(\phi^1) = \mathbf{R}(\phi^2)$ is satisfied only when $[\sigma_1^2, \boldsymbol{\theta}_1, \boldsymbol{\tau}_1^2, \boldsymbol{\alpha}_1] = [\sigma_2^2, \boldsymbol{\theta}_2, \boldsymbol{\tau}_2^2, \boldsymbol{\alpha}_2]$. For correlation functions satisfying Assumption 3.6, e.g., exponential family functions, the following lemmas show that each item in the covariance matrix \mathbf{R} is equal for two sets of parameters ϕ_1 and ϕ_2 only when $\phi_1 = \phi_2$, and hence, by Definition 3.5, we show identifiability.

Each item R_{ij} can be expressed as

$$R_{ij} = \begin{cases} G_{ij} + L_{ij}, & x_i, x_j \in \mathbf{D}_k \\ G_{ij}, & x_i \in \mathbf{D}_k, x_j \in \mathbf{D}_h, h \neq k \end{cases}, \quad (3.24)$$

where

$$G_{ij} = \begin{cases} \mathbf{g}_i \mathbf{G}_m^{-1} \mathbf{g}_j, & i \neq j \\ \sigma^2, & i = j \end{cases}, \quad L_{ij} = \begin{cases} \tau_k^2 \text{corr}_k^l(|x_i - x_j|, \boldsymbol{\alpha}_k) & i \neq j \\ \tau_k^2 & i = j \end{cases}.$$

The block diagonal items of \mathbf{R} is defined by the covariances between points in the same local region, while off the block diagonal items are the covariances between points in different local regions. Lemma 3.8 first shows that the off block diagonal items are the same only when the parameters are the same, and Lemma 3.9 shows similarly for the block diagonal items.

Lemma 3.8. *For all $x_i \in \mathbf{D}_p$ and all $x_j \in \mathbf{D}_q$ and $p \neq q$, the $(ij)_{th}$ element in covariance matrix $R_{ij}(\phi^1) = R_{ij}(\phi^2)$ is only satisfied when $\sigma_1^2 = \sigma_2^2$ and $\boldsymbol{\theta}_1 = \boldsymbol{\theta}_2$*

Lemma 3.9. *Given $\sigma_1^2 = \sigma_2^2$ and $\boldsymbol{\theta}_1 = \boldsymbol{\theta}_2$, for all $x_i, x_j \in \mathbf{D}_k$, $R_{ij}(\phi^1) = R_{ij}(\phi^2)$ is only satisfied when $\tau_1^2 = \tau_2^2$ and $\boldsymbol{\alpha}_1 = \boldsymbol{\alpha}_2$*

Theorem 3.10. *Under Assumptions 3.6 and 3.7, the set of parameters of the AGLGP model is identifiable.*

The detailed proof of these lemmas are provided in the Appendix D. As shown, the basic assumptions for piecewise independence of the local models provide the essential conditions for identifiability.

3.5 Numerical Experiments

In this section, numerical examples are presented to evaluate the performance of the model. First we investigate the efficiency of the selection of the inducing points and domain decomposition strategies. The AGLGP model is then numerically compared with the modified nugget effect kriging (MNEK) model (Yin et al., 2011) to study the approximation accuracy, and other approximation methods including full scale approximation method (FSA) (Sang and Huang, 2012), localized GP (LGP) model, reduced rank (RR) approximation (Banerjee et al., 2008) and PIC (Snelson and Ghahramani, 2007).

The implementations of all methods were conducted in MATLAB. As the implementations of the other methods are not available, we wrote our own codes for PIC, FSA, LGP and RR. Throughout the numerical analysis, we used the Gaussian covariance function. All numerical studies were performed on a processor with quad-core 3.3 GHz Core™ i5 CPU and 8 GB memory.

3.5.1 Effects of Inducing Points and Local Regions on AGLGP Model Estimation

In the proposed AGLGP model, a set of criteria are applied to generate inducing points and local regions. In this section, the effects of these criteria on the model estimation are compared with the popular criteria like k -means, Voronoi tessellation and grid partition. k -means (MacQueen, 1967) clustering technique aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean. Voronoi tessellation (Okabe et al., 2009) partitions the whole space into local regions based on their distance to a specified set of points, which is a specific subset of the space. There is a corresponding region for each of these points consisting of all points closer to that point than to any other. Grid partition is one of the simplest domain decomposition methods, which partitions the whole space into local regions by equally splitting each decision variable.

Here we consider a simple test function $y(x) = \cos(100(x - 0.2))e^{2x} + 7 \sin(10x)$ on $[0,1]$ with a noise function given by $\epsilon(x) \sim N(0, \sigma_\epsilon^2(x))$ and the noise variance is $\sigma_\epsilon^2(x) = 6 + 5 \sin(10x)$. Assume the domain space is decomposed into three local

regions. The average mean squared error (*IMSE*) between the AGLGP predictor $\hat{y}(x)$ (Equation 3.7) and the signal function $y(x)$ at N unsampled points works as the error measurement. In each macro replication k , we have $MSE(k) = \frac{1}{N} \sum_{i=1}^N (\hat{y}(x_i) - y(x_i))^2$. The average of *MSEs* based on M macro replications is given by

$$IMSE = \frac{1}{M} \sum_{k=1}^M MSE(k) \quad (3.25)$$

For a set of irregularly distributed design points, we first compare with k -means to generate different sets of inducing points given the same local regions. As shown in Table 3.1, given the same local regions, the AGLGP predictions with the proposed clustering method are significantly better than with k -means at level $\alpha = 0.05$. This Table 3.1: Error measurement of AGLGP model with different clustering techniques in local regions

AGLGP Clustering	k -means
1.5329	1.6595

is because the proposed clustering technique in AGLGP model not only considers the neighbouring locations as k -means but also considers the range of observation values to better capture the long lengthscale global trend.

Similarly, the AGLGP predictions are compared given different local regions which are generated by, for example, the regular grid partition, Voronoi tessellation and Support Vector Machine (SVM) (which is applied in the proposed AGLGP model estimation). The *IMSE* is compared given both a set of space-filling designs and a set of concentrated designs. The space-filling design is generated by Latin hypercube design while the concentrated design is generated based on a mixture of normal distributions $0.7N(1, 0.5^2) + 0.3N(4, 0.5^2)$.

Table 3.2 shows that the *IMSE* with SVM is significantly better than with the regular grid at $\alpha = 0.05$. Although there is no significant difference between the SVM and the Voronoi tessellation with space-filling designs, the SVM decomposition method is significantly better than the Voronoi tessellation with concentrated designs.

Table 3.2: Error measurement of AGLGP model with different region separation techniques

	SVM	regular grid	Voronoi
Space-filling designs	1.4981	1.5141	1.4962
Concentrated designs	6.6188	7.7606	6.9170

In general, the proposed approach to determine the local regions and inducing points works efficiently, especially for cases with concentrated designs like sequential optimization problems where the evaluated points tend to be clustered in promising local regions.

3.5.2 Comparative Studies for the AGLGP Model

In this section, AGLGP model is numerically compared with the modified nugget effect kriging (MNEK) model (Yin et al., 2011) and other approximation models including full scale approximation method (FSA) (Sang and Huang, 2012), localized GP (LGP) model, reduced rank (RR) approximation (Banerjee et al., 2008) and PIC (Snelson and Ghahramani, 2007).

We run two different test functions with different functional variability to compare the performance of the approximation models under different scenarios. Test 1 function is based on an irregular function $y = \cos(60(x - 0.1)) \exp(\sin(10x))$, with different local functional variability (short lengthscale) in different regions (Figure 3.4). Test 2 function is $y(x) = \sin(30(x - 0.9)^4) \cos(2(x - 0.9)) + (x - 0.9)/2$ as shown in Figure 3.5. This function has a long overall global trend (long lengthscale), and also local functional variation (short lengthscale).

We simulate output from these functions with three different noise level functions, namely $\sigma_e^1 = 0$, $\sigma_e^2 = 0.55 + 0.45 \sin(10x)$ and $\sigma_e^3 = 5.5 + 4.5 \sin(10x)$. We take 1000 design points and 20 replications are simulated at each design point. The average mean squared error between the predictor and the signal function at $N = 1000$ unsampled points is used as the error measurement. For FSA, PIC, and RR, we chose the same number of inducing inputs as AGLGP, which differs in each macro-replication. The

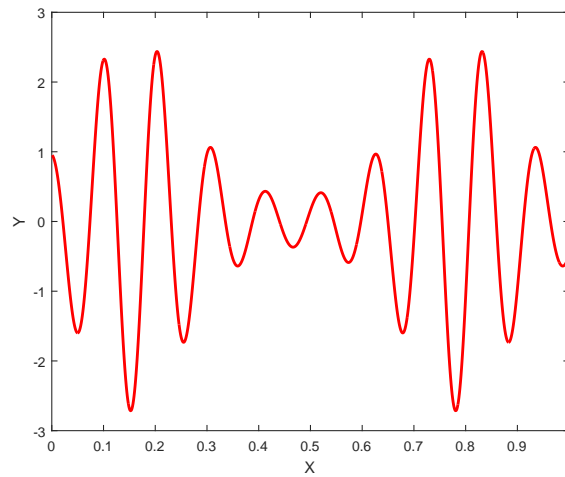


Figure 3.4: Test 1 function

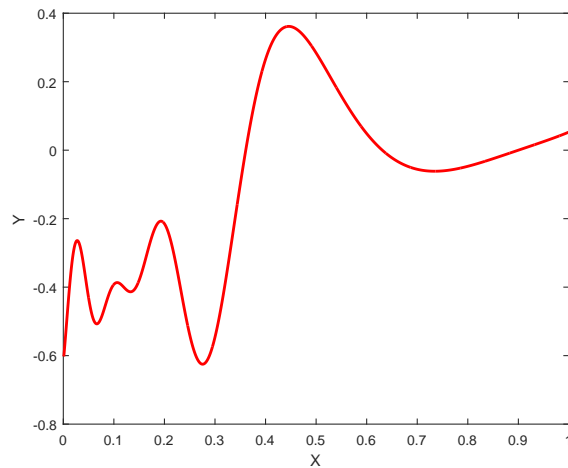


Figure 3.5: Test 2 function

tapering scale used for FSA is 0.2, and the same number of local regions is used for PIC and LGP as with AGLGP (equal to 5).

Table 3.3 summarizes the results. The AGLGP model and LGP model are statistically better than the other models (at $\alpha = 0.05$) for both test functions and at all levels of noise. For Test 1 where the function exhibits only short lengthscale local variation, the difference between all these models is not as large as the Test 2. For both test functions, the larger the noise variance, the smaller the relative difference between these models. In comparing the combined models, the AGLGP is able to outperform FSA and PIC as it is better able to capture the short lengthscale variability with independent local model parameters across the design space. Its performance is more pronounced in Test 2 where

Table 3.3: Error measurement of approximation models with one-dimension test function

Test 1	AGLGP	LGP	FSA	RR	PIC
$\sigma_e^1 = 0$	8.95×10^{-5}	8.99×10^{-5}	0.0589	0.0981	0.0605
$\sigma_e^2 = 0.55 + 0.45 \sin(10x)$	0.0031	0.0034	0.0951	0.1674	0.0793
$\sigma_e^3 = 5.5 + 4.5 \sin(10x)$	0.2046	0.2049	0.2859	0.3795	0.2919
Test 2	AGLGP	LGP	FSA	RR	PIC
$\sigma_e^1 = 0$	6.16×10^{-6}	4.16×10^{-6}	0.0020	0.0077	0.0048
$\sigma_e^2 = 0.55 + 0.45 \sin(10x)$	0.0006	0.0006	0.0028	0.0075	0.0022
$\sigma_e^3 = 5.5 + 4.5 \sin(10x)$	0.0301	0.0295	0.0405	0.0477	0.0408

the short lengthscale variability changes irregularly across different regions of the design space.

Furthermore, when comparing the computational times to estimate the model, the estimation of the LGP is the fastest, requiring 4.5623 seconds, followed by RR with 8.3598 seconds. AGLGP, FSA and PIC share the similar computational time of around 12 seconds.

Although the fitting of the LGP model is faster, it suffers larger discontinuities on the boundary as it is unable to capture correlation across local regions (off block diagonal), while the AGLGP captures it with a reduced rank like approximation. To study the extent of discontinuity of both models at the boundaries, we define the discontinuity measure as the absolute difference between the predictions from neighboring local models. Table 3.4 compared the average of the discontinuities over all boundaries for AGLGP model and LGP model. Results show that AGLGP model has smaller discontinuity for both functions at all noise levels.

Table 3.4: Discontinuity measurement of AGLGP and LGP models with one-dimension functions Test 1 and Test 2

	Test 1			Test 2		
	σ_e^1	σ_e^2	σ_e^3	σ_e^1	σ_e^2	σ_e^3
AGLGP	0.0699	0.0808	0.1450	0.0188	0.0247	0.0266
LGP	0.1093	0.1279	0.1937	0.0579	0.0755	0.1324

3.6 Conclusions

In this chapter, we proposed an additive global and local Gaussian Process (AGLGP) model to approximate the response surface of computer models where large data sets are observed. The proposed AGLGP model incorporates a global model and piecewise independent local models, which combine to capture the nonstationary features of the simulation models. The AGLGP model was also shown to be identifiable. The numerical study illustrated the performance of AGLGP with several other approximating GP models, and the results of AGLGP are promising. The global and local features of the AGLGP model make it promising in solving global optimization problems, especially in its application to combined global and local search algorithms.

Chapter 4

COMBINED GLOBAL AND LOCAL METHOD FOR STOCHASTIC SIMULATION OPTIMIZATION WITH AN AGLGP MODEL

4.1 Introduction

The development of the AGLGP model in Chapter 3 raises the interest of its application in optimization of stochastic simulation models. The AGLGP model is an additive surrogate that consists of a global model that captures the global trend and local models that capture the local residuals. With its global and local structure and computational efficiency, the AGLGP model is also well suited for simulation optimization. In this chapter, we develop a new optimization algorithm that leverages on the global and local structure for a combined iterative global and local search.

Given an objective function $f(x)$, where the design points x have a feasible region \mathbb{X}_Ω , the general target of global optimization is to find the optimal solution x^* that satisfies $f(x^*) = \min_{x \in \mathbb{X}_\Omega} f(x)$. In stochastic simulation, as the objective function can

only be observed with noise, we are interested in solving the following problem:

$$\min_{x \in \mathbb{X}_\Omega} E(y(x)) \quad (4.1)$$

where $y(x)$ are sample observations.

To solve such problems, there are generally two classes of approaches: direct methods and metamodel-based methods. Popular direct methods include adaptive random search such as COMPASS (Hong and Nelson, 2006), nested partition (NP) methods (Shi and Ólafsson, 2000), and other heuristic methods such as genetic algorithm and simulated annealing approaches. These direct methods are applied directly into the simulator and the search is conducted on the simulator. COMPASS provides an efficient method to find local optimal solutions in a stochastic environment. NP systematically partitions the feasible region into subregions, assesses the potential of each region, and then concentrates the computational effort in the most-promising region. Some direct methods have been shown to be globally convergent, but they typically require a larger number of simulations to obtain adequately good solutions.

Metamodel based methods estimate a metamodel (also known as *surrogate model*, or *response surface model*) with few simulations in the search procedure, to quickly evaluate the performance at any given point in the domain space without the need to run the simulator at every potential point. Such methods provide the information of the entire surface for better identifying the points for further simulation. The Gaussian Process (GP) model has been used in various types of black-box optimization problems. A review of global optimization algorithms for deterministic computer models has been conducted by Jones (2001). The Efficient Global Optimization (EGO) algorithm (Jones et al., 1998) is a popular and widely used algorithm that determines the next evaluation point by maximizing the expected improvement (EI) from the current optimal solution. Huang et al. (2006), Picheny et al. (2013), and Quan et al. (2013) further extended this framework to stochastic simulation with homogeneous or heterogeneous variance.

Sequential Kriging Optimization (SKO) proposed by Huang et al. (2006) is adaptable to the stochastic simulation model. With an augmented EI function, the SKO accounted for the influence of random noise $\epsilon(x)$ and considered the option of adding replications

on the existing design points other than exploration for new design points. However, the SKO only considered the homoscedastic cases, assuming that the random noise function has constant variability throughout the entire sample space. Picheny et al. (2013) proposed an expected quantile improvement (EQI), taking the kriging percentile as a measure. It works well for heterogeneous noise, but the noise variance has to be known. Quan et al. (2013) developed a two stage sequential optimization (TSSO) framework that can be adopted for the optimization of stochastic simulation with heterogeneous variances. Without additional requirement for the noise variance function, the algorithm is less computational demanding. However, all of these methods may not work efficiently when optimizing a highly nonstationary response surface, where a large number of evaluations might be required to get a reasonably good solution.

In this chapter, we propose a combined global and local optimization (CGLO) algorithm based on the AGLGP model proposed in Chapter 3. Specifically, as the global model in AGLGP model captures the global (long lengthscale) trend, it is useful in identifying promising regions to conduct more detailed search. Within a promising region where the trend is low, the AGLGP model with its local information can be further used to search more finely for a best minimum point in that region. Finally, as a local model provides information on the residuals, it can be applied to determine the noise and bias in each region. Our algorithm shares a similar framework with the Efficient Global Optimization (EGO) (Jones et al., 1998), but works more efficiently due to the global and local features of the AGLGP model, when the number of evaluations observed gets large as the iteration progresses to find the optimal solution.

This chapter is organized as follows. In Section 4.2, we first review the background of the expected improvement functions. The detailed development of the CGLO algorithm is provided in Section 4.3. Section 4.4 presents its convergence properties. Numerical results are given in Section 4.5 and Section 4.6 followed by conclusions in Section 4.7.

4.2 The Expected Improvement Function and the Modified Expected Improvement Function

In surrogate-based optimization algorithms, several criteria of selecting the next evaluation point such as minimizing the response, minimizing a lower bound of the response, maximizing the probability of improvement or maximizing the expected improvement, have been proposed (Jones, 2001). Among these criteria, the expected improvement distinguishes itself for its ability in automatically balancing between exploration and exploitation to efficiently find the global optimal solution. Here we review the expected improvement function and its extensions in the stochastic case with heterogeneous noise variance.

Jones et al. (1998) defined the improvement function at point x_0 as $I(x_0) = \max[f_{min} - f(x_0), 0]$ where f_{min} is the observed optimal solution. At any unsampled point x_0 , the mean response $f(x_0)$ is unknown, but can be predicted with a mean $\hat{f}(x_0)$ and a variance $\hat{s}^2(x_0)$, and hence the expectation is considered.

$$E[I(x_0)] = E[\max[f_{min} - f(x_0), 0]] \quad (4.2)$$

This criterion not only considers the point with lower predicted response surface, but also with high spatial uncertainty in the response. According to Jones et al. (1998), Equation (4.2) can be computed by

$$E[I(x_0)] = (f_{min} - \hat{f}(x_0))\Phi\left(\frac{f_{min} - \hat{f}(x_0)}{\hat{s}(x_0)}\right) + \hat{s}(x_0)\phi\left(\frac{f_{min} - \hat{f}(x_0)}{\hat{s}(x_0)}\right) \quad (4.3)$$

Quan et al. (2013) adopted the EI function to address the stochastic simulation systems with heterogeneous variance. They adopted the modified nugget effect kriging (MNEK) model (Yin et al., 2011) and proposed the following modified expected improvement (mEI) to drive the sequential search for new evaluation points in the search stage.

$$mEI(x) = E(\max[y_{min} - y(x), 0]) \quad (4.4)$$

In this proposed criterion, y_{min} is the predicted response at the sampled point with the lowest sample mean, and $y(x)$ is a normal random variable with mean given by the MNEK predictor at x as shown in Equation (3.2) and variance given by the predictors spatial prediction uncertainty

$$\hat{s}_z^2(x_0) = E(f(x_0) - \hat{y}(x_0))^2 = \sigma^2 - \mathbf{r}'\mathbf{R}^{-1}\mathbf{r}. \quad (4.5)$$

By ignoring predictor uncertainty caused by random variability, the mEI function assumes that the observations are made with infinite precision so the same point is never selected again. This enables the search to focus on new points in promising regions with high predicted responses and new points that reduce the spatial uncertainty of the metamodel.

4.3 Development of Methodology

4.3.1 General Framework of the CGLO Algorithm

The CGLO algorithm is an iterative algorithm composed of a global search stage followed by a local search stage. The global search stage exploits the whole space globally while the local search stage exploits within a promising region to find the optimal point in that region. Local search here emphasizes the search in the promising local region. The terminology is not to be confused with the search for a local solution to the optimization problem here. Considering a total budget of T , in each iteration t , the budget exhausted is denoted as B_t , where $B_t = B_{t,s} + B_{t,a}$. The budget $B_{t,s}$ is for the local search step to find new design points and the budget $B_{t,a}$ is for the local allocation step to allocate replications to existing design points. n_{max} is the maximum number of new evaluation points at each iteration. An overview of the algorithm is given in Table 4.1.

In the next subsections, we describe Steps 3 and 4 in detail. It is worthwhile to note that the local regions are decided based on the space-filling initial designs and are kept constant over iterations. This is to facilitate computation by avoiding the redivision of regions after each iteration. In optimization problems where our ultimate goal is global optimization, most of the design points tend to end up in several clusters. If one of the

Table 4.1: Overview of CGLO

Combined Global and Local Optimization Algorithm	
<i>Step 1:</i>	<i>(Initialization)</i> Run a size n_0 space filling designs, with r_{min} replications allocated to each point. Total initial replications $B_0 = n_0 r_{min}$. Set $t = 0$.
<i>Step 2:</i>	<i>(Validation of overall model)</i> Fit an AGLGP response model to the set of sample means and variances, and use cross validation to ensure that the AGLGP prediction is satisfactory.
While the available replications $A = T - \sum_{i=0}^t B_i > 0$, $t = t + 1$	
<i>Step 3:</i>	<i>(Global Search)</i> Generate n_c candidate locations Ω_g . Select a point x_0^g among Ω_g based on the global model and a global criterion, and identify the local region $\mathbf{D}_{k,t}$, where $x_0^g \in \mathbf{D}_{k,t}$ to conduct the local search.
<i>Step 4.1:</i>	<i>(Local Search Step)</i> While $n_t < n_{max}$ and $A > 0$, <i>(Fit/Update a local model)</i> Fit or update the local model in the local region $\mathbf{D}_{k,t}$. <i>(Generate Candidate Points)</i> Randomly generate n_l candidate points $\Omega_l \in \mathbf{D}_{k,t}$. <i>(Select the Next Evaluation Point)</i> Search for the location $x_{n_t}^*$ from the candidate points Ω_l based on the overall model and a local search criterion and evaluate at $x_{n_t}^*$ with r_{min} replications. Break Step 4.1 if a switching criterion is satisfied. end
<i>Step 4.2:</i>	<i>(Local Allocation Step)</i> Decide the budget $B_{t,a}$ and the allocation strategy for additional evaluations among those evaluated points in $\mathbf{D}_{k,t}$. end
<i>Step 5:</i>	<i>(Return the Optimal Solution)</i> Return the point with the lowest sample mean.

clusters ends up located near a boundary, and the observations in the cluster are dense, a reasonably good model can be developed already with the information from only these dense points without considering the correlations from the neighboring region. If however, only few points are located near the boundary, the independence assumption will hold true. Hence, keeping local regions constant throughout the algorithm is approximately reasonable. The algorithm can also be modified to update the local regions after Step 4.2 in each iteration.

4.3.2 Global Search Stage

In the global search in Step 3, CGLO focuses on identifying a promising local region to focus the search. No simulation or budget is exhausted in this stage. The promising region is identified based on the estimated global model only. The global search stage randomly generates a set of candidate points Ω_g and selects the point x_0^g that maximizes the global expected improvement among Ω_g . The local region \mathbf{D}_k where $x_0^g \in \mathbf{D}_k$ will be selected as the promising region for further local search. This sampling procedure is similar to the Stochastic Response Surface (SRS) framework (Regis and Shoemaker, 2007).

4.3.2.1 Generation of Candidate Points

Here we provide some details on the generation of candidate points in step 3. Suppose we generate n_c candidate points as Ω_g . The following condition will be imposed on the selection of candidates. [C1] The candidate points Ω_g have to be spread out across the whole space and the size must be at least equal to the number of local regions such that there exists at least one candidate point in each local region. Possible choices include a Latin Hypercube design across the regions and Treed partitioning that divides up the input space by making binary splits on the value of each variable (Breiman et al., 1984). Based on our numerical tests, we find that an n_c value of about $10k$ points is a reasonable choice for a k dimensional problem.

4.3.2.2 Global Expected Improvement

As the purpose of the global search is to quickly identify a promising local region to concentrate the local search, we want the algorithm in this stage to explore the whole design space for promising solutions in regions that have not been sufficiently searched (in the hope of finding regions better than the regions that have been searched so far), and in regions with potentially good minimum solutions.

Based on the global model which smooths out the localized features in each local region, the predictive global trend can provide a guide towards the promising solutions in the design space. However, as the global model is based on inducing points, the predictive variance $\hat{s}_g^2(x)$ does not directly reflect the spread of the observations. Specifically, a high $\hat{s}_g^2(x)$ can indicate few or no inducing points in that region. In regions where there are no inducing points, there are no observations and hence exploration should be done in that region, while few inducing points in a region can imply two possibilities on the distribution of observations in that region. If the few inducing points are due to few observations, it is worth exploring further in that region. However, if the few inducing points are due to a large number of observations in that region being very similar (small variability among them) and hence clustered together, thus requiring only one inducing point to closely represent them, there is little need to further explore there as it is highly explored with observations that are all close together. Hence, to balance the global search, the number of observations in each region is another important indicator to ensure a spread of observations, especially in promising regions.

In order to achieve the goals of identifying regions that are less explored and also regions with promising low trend in this global stage to enable the local stage to pursue more intensely, we propose to use a modified global EI function,

$$gEI(x) = E\{\max(y_{gmin} - y_g(x), 0)\} \cdot \frac{1}{1 + e^{n_i/v-5}} \quad (4.6)$$

where y_{gmin} is the lowest predicted global evaluation among the current inducing points. $y_g(x)$ is a normal random variable with mean given by $\hat{y}_g(x)$ and variance given by $\hat{s}_g^2(x)$. The first term of the product in Equation (4.6) is the EI function which has incorporated the location of the inducing points and the global trend and represents how much a

new point is expected to be better than the current best solution of the global model. The second term is a factor designed to account for observations around each point that have been aggregated away with the inducing points. It serves as a penalty for points with many observations around it, giving diminishing returns overall for points with increasing number of observations around it. Specifically, the factor is a logistic function that decreases as n_i increases, where n_i is the number of neighbors of candidate point x_c^i . v is a user-defined parameter that controls the steepness of the function, which represents how fast the gEI decreases with the increase of observations. The set of neighbors of $x_c^i \in \mathbf{D}_k$ is defined as $\mathbb{B}(x_c^i) = \{x \in \mathbf{x}_1^k : \|x - x_c^i\| < \|x - x_c^j\|, \forall x_c^j \in \Omega_g \cap \mathbf{D}_k, j \neq i\}$. With given n_i , the larger value of v , the larger the factor. The factor approaches zero when $n_i/v \geq 10$, and so a possible choice for v is $v = MAX/(10n_c)$, where MAX is the maximum number of evaluation points allowed given the total budget and n_c is the number of global candidate points.

Overall the gEI in Equation (4.6) is consistent with our desire to have the global stage identify regions that are promising (in terms of lesser explored and hence possess the potential of a better solution, and lower global trend which may also yield a better solution), and the point obtained by maximizing Equation (4.6) will indicate the promising region to search more comprehensively in the local stage.

4.3.3 Local Search Stage

Next we provide the details on the local search. Once the candidate point x_g^0 is selected by the global search, the local region \mathbf{D}_k in which x_g^0 lies will be selected as the promising region. The local stage then searches more intensely within this promising region for a better solution. Here we adopt a similar approach to TSSO (Quan et al., 2013), in which this local stage is divided into a search step to find new design points and an allocation step to manage the random variability within this region. A budget $B_{t,s}$ is allocated to the search step and a budget $B_{t,a}$ is allocated to the allocation step. We describe the details of this local search step (Step 4.1 in Table 4.1) and local allocation step (Step 4.2 in Table 4.1) in the following subsections.

4.3.3.1 Local Search Step

The role of this local search step is to start from the local area around the promising point x_g^0 identified in the global stage and quickly progress to a better solution within the local region. In order to achieve this, we apply the mEI criterion (Quan et al., 2013) to sequentially select the next evaluation point,

$$mEI(x) = E(\max[y_{min} - z(x), 0]), x \in \Omega_l$$

where y_{min} is the predicted response at the sampled points in the promising region with the lowest sample mean, and $z(x)$ is a normal random variable with mean given by AGLGP predictor and variance given by spatial prediction uncertainty $\hat{s}_z^2 = L_{nn} - \mathbf{1}'\mathbf{L}_n^{-1}\mathbf{1}$. Here the overall AGLGP model is used as we require a more comprehensive and accurate search within this region for a better minimum point. By ignoring predictor uncertainty caused by random variability, the mEI assumes that the observations are made with infinite precision so the same point is never selected again (as the local allocation step will by design address the stochastic noise). This enables the search to focus on new points with low predicted response and new points in the less explored areas of the local region. The point $x^* \in \Omega_l$ that maximizes the mEI will be simulated with r_{min} replications. The candidate points Ω_l are randomly generated from a normal distribution with mean x_g^0 and variance σ_n^2 . Here, σ_n^2 is defined by the minimum distance between global candidate points to make the majority of the local candidates Ω_l generated within the neighborhood of x_g^0 .

The local search step is sequentially repeated until a switching criterion is satisfied. In each repetition of this local step, the candidate set of points Ω_l is updated as the search sequentially searches around the local region, and is taken to consist of normally generated points in the neighborhood of the last point simulated x^* . The overall budget expended in this local search step is $B_{t,s} = n_t \times r_{min}$, where n_t is the number of iterations (or points evaluated) in this search step. As n_t varies from iteration to iteration of the algorithm, n_t and hence $B_{t,s}$ are dynamically decided in each iteration of the algorithm.

Switching Criterion Each time a new evaluation point is simulated, the local model and y_{min} will be updated accordingly. The points in the local region \mathbf{D}_k will be reclustered and inducing points updated. As more new points are observed, the number of inducing points and their values in this region can significantly change, potentially changing the global model. The switching criterion of the local stage then dynamically decides whether to continue with the local search in the current local region or return to a global search in the entire design space. As the global search employs the global model in the gEI, it is reasonable to return to the global search again when either the global model changes or the penalty factor in the gEI reduces to identify a new promising region.

To prevent selecting the same promising region in consecutive iterations, we require the algorithm to return to the global search stage only when the number of inducing points in the promising region increases or a maximum number of new design points n_{max} that gives a smaller gEI in the current promising region than the other regions,

$$\max_{x_c \in \Omega_g \cap \mathbf{D}_k} gEI(x_c) \leq \max_{x_c \in \Omega_g \cap \mathbf{D} \setminus \mathbf{D}_k} gEI(x_c), \quad (4.7)$$

is met. This reduces the number of iterations between global and local search. The refitting of the global model is only done when the global model or gEI changes. By defining the switching criterion, we also limit the budget exhausted in this step to $n_{max} \times r_{min}$. If the remaining budget is smaller than this limit ($T - \sum_{i=1}^{t-1} B_i < n_{max} \times r_{min}$), additional simulations should focus on the allocation step for finding the optimal solution.

4.3.3.2 Local Allocation Step

The allocation step is dedicated to distribute additional simulations among sampled points to improve the evaluation of the noisy simulations. For every evaluated point, more simulation replications can be allocated for two purposes. Firstly, to improve the fit of the global model (for a more efficient global search), more simulation replications should be allocated to improve the clustering and the estimation of inducing points. Secondly, to improve the local model estimation and the local optimal value (i.e., optimal in the local region), more simulation replications are needed to improve the observations

4.3. DEVELOPMENT OF METHODOLOGY

at evaluated points. In this step, we dynamically decide the budget allocated by taking the maximum of budget allocated to each point that satisfies both the purposes.

First, to improve the clustering and estimation of inducing points within a local region, the effect of random noise on the clustering technique has to be evaluated. An evaluation point might get wrongly clustered based on the defined contour lines. Given the contour lines, we define that a point is wrongly clustered if its noisy observation falls into a different cluster from its 'best' cluster, defined as the cluster where its true mean value (first two terms of Equation (3.4)) falls into. To decide which evaluation point is likely to be wrongly clustered and how much budget should be allocated to bring it to its 'best' cluster, an evaluation criterion related to the distance to the contour lines and the noise variance is used. This is explained by Equation (4.8) and Equation (4.9) below.

An observation with a large noise variance or a small distance to the contour lines has a high chance of being wrongly clustered. The noise variance only decreases at the rate of $1/\sqrt{r}$ when averaging over r replications, but we can be at least approximately 99.7% confident that one point x_i is rightly clustered if

$$\Delta_i \geq 3\hat{\sigma}_\epsilon^2(x_i)/\sqrt{N_{i,v} + r_i} \quad (4.8)$$

where Δ_i is its response distance to the nearest contour line, $\Delta_i = \min_{y \in \mathbb{Y}} |\bar{y}(x_i) - y|$, $\hat{\sigma}_\epsilon^2(x_i)$ is sample variance estimated through r_i replications and $N_{i,v}$ is the number of additional simulations required for improving the cluster where the evaluation point x_i lies. Hence, $N_{i,v} \geq (3\hat{\sigma}_\epsilon^2(x_i)/\Delta_i)^2 - r_i$. As we do not want to allocate too much budget without the updating information, the budget allocated to improve the clusters is limited by a maximum number v_{min} . Specifically,

$$N_{i,v} = \begin{cases} \min\{[(3\hat{\sigma}_\epsilon^2(x_i)/\Delta_i)^2] - r_i, v_{min}\} & \text{for } (3\hat{\sigma}_\epsilon^2(x_i)/\Delta_i)^2 > 1 \\ 0 & \text{for } (3\hat{\sigma}_\epsilon^2(x_i)/\Delta_i)^2 \leq 1 \end{cases} \quad (4.9)$$

For the second purpose of improving the local optimal value, additional replications are distributed with the goal of maximizing the probability of the correct selection of the local optimum. The maximum number of replications for this purpose is denoted as $B_{t,b}$. OCBA provides a rigorous way of allocating budget to identify the sampled

point with the best response by balancing the response and noise level observed for each point. Suppose each point x_i has a sample mean given by \bar{y}_i and a sample variance $\sigma_\epsilon^2(x_i)$. Then according to Theorem 3.2 provided by Chen and Lee (2010), the Approximate Probability of Correct Selection (APCS) can be asymptotically maximized when available budget tends to infinity and when

$$\frac{N_{i,b}}{N_{j,b}} = \left(\frac{\sigma_\epsilon(x_i)/\Delta_{b,i}}{\sigma_\epsilon(x_j)/\Delta_{b,j}} \right)^2, i, j \in \{1, 2, \dots, r_{k^*}\}, i \neq j \neq b \quad (4.10)$$

$$N_{b,b} = \sigma_\epsilon(x_b) \sqrt{\sum_{i=1, i \neq b}^n \left(\frac{N_{i,b}}{\sigma_\epsilon(x_i)} \right)^2}$$

where \bar{y}_b is the lowest sample mean in the current region, $N_{i,b}$ is the number of simulations allocated to point x_i and $N_{b,b}$ is for x_b with the lowest sample mean, so $\sum_i N_{i,b} = B_{t,b}$. $\Delta_{b,i} = \bar{y}_i - \bar{y}_b$. As the focus here is on illustrating the application of the AGLGP model in optimization, we assign constant $B_{t,b}$ over iterations for simplicity. More sophisticated budget allocation strategies can be applied to improve the convergence rate.

Finally we have $N_i = \max(N_{i,v}, N_{i,b})$ to satisfy the two purposes of improving the estimation of inducing points and improving the global optimal solution in the local region. Also $B_{t,a} = \sum_{j=1}^{r_i} N_j$. We note here that the allocation budget required is dynamically decided in each iteration based on the number required to improve the estimation of the inducing points and the number required to improve the estimation of the local model. If the remaining budget $A = T - \sum_{i=1}^{t-1} B_i - B_{t,s} < B_{t,a}$, it is reasonable to allocate the remaining budget to find the best optimal solution using the OCBA allocation only as no additional iterations will be available to refit the global model for global search any more.

4.4 Convergence of the CGLO Algorithm

To prove the global convergence of the CGLO algorithm, we first state the following lemmas. Besides the condition [C1] proposed in section 4.3.2.1, we further add an additional condition [C2] to facilitate the proof. [C2] Ω_g is constant over iterations. This enables the EI type function for the global search to be comparable across iterations as it moves from one global candidate point to another between iterations. This adds on

4.4. CONVERGENCE OF THE CGLO ALGORITHM

to condition [C1] described in Section 6.1 which ensures that there exists at least one candidate point in each local region such that each region will possibly be explored.

Suppose the objective function f has a unique global optimal $f^* = f(x^*) = \min_{x \in \mathbb{X}_\Omega} f(x)$ at x^* over the whole domain \mathbb{X}_Ω . We denote evaluated points up to iteration t as \mathbb{X}_t . The global search iteratively selects x_0^g among Ω_g , while the local search iteratively selects evaluation points in the neighborhood of x_0^g . For $x_0^g \in \mathbf{D}_k$, the neighborhood is defined as $\mathbb{B}(x_0^g) = \{x \in \mathbf{D}_k : \|x - x_0^g\| < \|x - x_c\|, \forall x_c \in \Omega_g \cap \mathbf{D}_k, x_c \neq x_0^g\}$. So $\cup_{x_c \in \Omega_g \cap \mathbf{D}_k} \mathbb{B}(x_c) = \mathbf{D}_k$ and $\cup_{x_c \in \Omega_g} \mathbb{B}(x_c) = \mathbb{X}_\Omega$.

Lemma 4.1. $\forall x^g \in \Omega_g$, x^g will be selected infinitely often (i.o.) by global search.

Proof. The global predictive mean and variance are estimated as

$$\begin{aligned}\hat{y}_g(x) &= \beta_0 + \mathbf{g}' \mathbf{Q}_m^{-1} \mathbf{G}_{mn} (\boldsymbol{\Lambda} + \boldsymbol{\Sigma}_\epsilon)^{-1} (\mathbf{y} - \mathbf{1}' \beta_0) \\ \hat{s}_g^2(x) &= [G_{nn} - \mathbf{g}' \mathbf{G}_m^{-1} \mathbf{g}] + \mathbf{g}' \mathbf{Q}_m^{-1} \mathbf{g}\end{aligned}$$

where $\forall x^g \in \Omega_g$, $\hat{s}_g^2(x^g) > 0$, as $\hat{y}_g(x)$ is estimated through the noise observation \mathbf{y} and $\hat{s}_g^2(x^g)$ is inflated by the random noise. Hence, $EI(x^g) = E\{\max(y_{gmin} - y_g(x), 0)\} > 0$, where y_{gmin} is the lowest predicted global evaluation among the current inducing points. It suffices to prove for all $x^g \in \Omega_g$ and any $\epsilon > 0$, $gEI(x^g) \leq \epsilon$ is achieved in a finite number of iterations, because if $\exists x_0^g \in \Omega_g$ being selected finite number of times, there exists $\epsilon_1 > 0$ that satisfies $gEI(x_0^g) \geq \epsilon_1$. We note that

$$\begin{aligned}EI(y_g(x)) &= \int_{-\infty}^{y_{gmin}} (y_{gmin} - y_g(x)) dF(y_g(x)) \\ &= \hat{s}_g(x) \Psi\left(\frac{\hat{y}_g(x) - y_{gmin}}{\hat{s}_g(x)}\right) \leq \frac{\hat{s}_g(x)}{\sqrt{2\pi}}\end{aligned}\tag{4.11}$$

where $y_g(x) \sim N(\hat{y}_g(x), \hat{s}_g^2(x))$ and $\Psi(t) = \psi(t) + t\phi(t)$. ψ is the standard normal density and ϕ is the standard normal distribution. Given

$$gEI(x) = E\{\max(y_{gmin} - y_g(x), 0)\} \cdot \frac{1}{1 + e^{n_i/10-5}}\tag{4.12}$$

for any $\epsilon > 0$, $gEI(x^g) \leq \epsilon$ is satisfied when its upper bound is smaller than ϵ . Then we have n_i is bounded from n_i^*

$$\frac{\hat{s}_g(x)}{\sqrt{2\pi}} \cdot \frac{1}{1 + e^{n_i/10-5}} \leq \epsilon \quad (4.13)$$

$$n_i \geq n_i^* = 10 \left(\ln \left(\frac{\hat{s}_g(x)}{\epsilon\sqrt{2\pi}} - 1 \right) + 5 \right) \quad (4.14)$$

For any $x^g \in \Omega_g$, $gEI(x^g) \leq \epsilon$ is satisfied by a finite number of iterations $t^* = \sum_{x_i^g \in \Omega_g} n_i^*$ \square

In the next lemma, we show that since any $x^g \in \Omega_g$ is selected *i.o.*, the points in the neighborhood of x^g will be dense by local search, and hence the set of \mathbb{X}_t will be dense in the whole space.

Lemma 4.2. *If any $x \in \Omega_g$ is selected infinitely often (i.o.) by global search, the set of points observed \mathbb{X}_t is dense in \mathbb{X}_Ω as $t \rightarrow \infty$*

Proof. To prove the set of points observed \mathbb{X}_t is dense in \mathbb{X}_Ω , it suffices to prove that each local region \mathbf{D}_k is dense since $\cup_k \mathbf{D}_k = \mathbb{X}_\Omega$. Based on Lemma 4.1, each global candidate is selected infinitely often, hence each local region is allocated with infinite budget.

As we follow the SRS framework by Regis and Shoemaker (2007) in the local search stage, which generates candidate points from the normal distribution centered at the last simulated point, from Theorem 1 in Regis and Shoemaker (2007), the sequence of observed global optimizers will converge to the true global optimizer $x_t^* \rightarrow x^*$ almost surely. Hence the observed global optimizer in each local region \mathbf{D}_k will converge to the true global optimizer in \mathbf{D}_k .

Furthermore, from Theorem 1.3 in Torn and Zilinskas (1989) that is restated below,

Theorem. 1.3 (Torn and Zilinskas, 1989) *Let the minimization region A be compact. Then a global minimization algorithm converges to the global minimum of any continuous function iff the sequence of trial points of the algorithm is everywhere dense in A .*

we have dense local region \mathbf{D}_k for $\forall k$. \square

4.4. CONVERGENCE OF THE CGLO ALGORITHM

Given Lemma 4.1, we have that each region \mathbf{D}_k will be selected as a promising region *i.o.*. In the next lemma, we show that based on Lemma 4.2, each point in \mathbf{D}_k will be allocated with infinite replications. Hence each point in \mathbb{X}_Ω will be allocated with infinite replications. Then by strong law of large numbers, the average of sample means will converge to the true mean, and we have Lemma 4.3.

Lemma 4.3. *For any $\epsilon > 0$, $\forall x \in \mathbb{X}_\Omega$, $P\{|\bar{y}_t(x) - f(x)| > \epsilon, i.o.\} = 0$*

where $\bar{y}_t(x)$ is the sample mean of an observed point x at iteration t .

Proof. This is equivalent to show that $\forall x \in \mathbb{X}_\Omega$, $r_t(x) \rightarrow \infty$ with probability 1 (*w.p.1*), where $r_t(x)$ the number of replications allocated to x at iteration t . Then by strong law of large numbers, we have Lemma 4.3. We assume a constant evaluation budget allocated by OCBA in each iteration, so the total amount of budget will go to infinity as iteration goes to infinity. From the results of Theorem 3.2 in OCBA (Chen and Lee, 2010), each point will be selected as the best point *i.o.*. Hence $r_t(x) \rightarrow \infty \forall x \in \mathbb{X}_\Omega$

□

With these lemmas, we have the main results of convergence for the CGLO algorithm.

Let $\bar{y}_t^* = \min_{x \in \mathbb{X}_t} \bar{y}_t(x)$ and x_t^* is the optimal solution by iteration t .

Theorem 4.4. *Suppose the CGLO is used to solve the optimization problem (4.1), $\bar{y}_t^* \rightarrow f^*$ and $x_t^* \rightarrow x^*$ *w.p.1* as $t \rightarrow \infty$*

Proof. Notice that

$$\begin{aligned} |\bar{y}_t^* - f^*| &= \left| \min_{x \in \mathbb{X}_t} \bar{y}_t(x) - \min_{x \in \mathbb{X}_\Omega} f(x) \right| \\ &\leq \left| \min_{x \in \mathbb{X}_t} \bar{y}_t(x) - \min_{x \in \mathbb{X}_\Omega} \bar{y}_t(x) \right| + \left| \min_{x \in \mathbb{X}_\Omega} \bar{y}_t(x) - \min_{x \in \mathbb{X}_\Omega} f(x) \right| \\ &\leq \left| \min_{x \in \mathbb{X}_t} \bar{y}_t(x) - \min_{x \in \mathbb{X}_\Omega} \bar{y}_t(x) \right| + \max_{x \in \mathbb{X}_\Omega} |\bar{y}_t(x) - f(x)| \\ &= \max_{x \in \mathbb{X}_\Omega} |\bar{y}_t(x) - f(x)| \end{aligned}$$

where the last equation follows from Lemma 4.2. Therefore, for any $\epsilon > 0$,

$$P\{|\bar{y}_t^* - f^*| > \epsilon, i.o.\} = P\left\{ \left| \min_{x \in \mathbb{X}_t} \bar{y}_t(x) - \min_{x \in \mathbb{X}_\Omega} f(x) \right| > \epsilon, i.o.\right\}$$

$$\leq P\{\max_{x \in \mathbb{X}_\Omega} |\bar{y}_t(x) - f(x)| > \epsilon, i.o.\}$$

Given Lemma 4.3, $P\{\max_{x \in \mathbb{X}_\Omega} |\bar{y}_t(x) - f(x)| > \epsilon, i.o.\} = 0$. We conclude $\bar{y}_t^* \rightarrow f^*$ w.p.1 as $t \rightarrow \infty$.

Next, we are showing $f(x_t^*) \rightarrow f(x^*)$ w.p.1 as $t \rightarrow \infty$, and hence the uniqueness of x^* implies $x_t^* \rightarrow x^*$ w.p.1 as $t \rightarrow \infty$. Notice that $|f(x_t^*) - f(x^*)| \leq |f(x_t^*) - \bar{y}_t(x_t^*)| + |\bar{y}_t(x_t^*) - f(x^*)|$, hence

$$P\{|f(x_t^*) - f(x^*)| > 2\epsilon, i.o.\} \leq P\{|f(x_t^*) - \bar{y}_t(x_t^*)| + |\bar{y}_t(x_t^*) - f(x^*)| > 2\epsilon, i.o.\}$$

As $\{|f(x_t^*) - \bar{y}_t(x_t^*)| + |\bar{y}_t(x_t^*) - f(x^*)| > 2\epsilon\} \subseteq \{|f(x_t^*) - \bar{y}_t(x_t^*)| > \epsilon\} \cup \{|\bar{y}_t(x_t^*) - f(x^*)| > \epsilon\}$, we have

$$\begin{aligned} P\{|f(x_t^*) - \bar{y}_t(x_t^*)| + |\bar{y}_t(x_t^*) - f(x^*)| > 2\epsilon\} \\ \leq P\{|f(x_t^*) - \bar{y}_t(x_t^*)| > \epsilon\} + P\{|\bar{y}_t(x_t^*) - f(x^*)| > \epsilon\}. \end{aligned}$$

By Lemma 4.3, we have $P\{|\bar{y}_t(x_t^*) - f(x_t^*)| > \epsilon, i.o.\} = 0$ and we have shown that $P\{|\bar{y}_t(x_t^*) - f(x^*)| > \epsilon, i.o.\} = 0$. So we can see that $P\{|f(x_t^*) - f(x^*)| > 2\epsilon, i.o.\} = 0$ and we conclude that $f(x_t^*) \rightarrow f(x^*)$ w.p.1 as $t \rightarrow \infty$. The uniqueness of x^* implies $x_t^* \rightarrow x^*$ \square

The convergence is proved under condition [C2]. When [C2] is relaxed and we allow for random generated global candidate points Ω_g across iterations, we conjecture that the search can still be dense in the whole design space as long as for $\forall x \in \mathbb{X}_\Omega$, the neighborhoods $\mathbb{B}(x_g^0)$ where $x \in \mathbb{B}(x_g^0)$ will be selected for local search infinitely often.

4.5 Numerical Results

In this section, we illustrate the proposed CGLO algorithm, compare it with other surrogate optimization algorithms in both efficiency and computational time.

4.5.1 One-dimensional Test Function (Illustration of Algorithm)

We applied the proposed algorithm to the one-dimensional example $\cos(100(x - 0.2)) \exp(2x) + 7 \sin(10x)$ with the noise variance function $0.2 + 0.1 \sin(10x)$ to further illustrate how the algorithm works. We initialize 12 design points with a Latin Hypercube design and 3 local regions, as seen in Figure 4.1. The test function is derived to have a global optimal solution -10.1316 at $x = 0.9865$ and another sub-optimal solution -9.5799 at $x = 0.4826$. Hence, regions 2 and 3 are more promising regions with lower response that should be focused on. Each design point is evaluated with a minimum of 20 replications and 10 additional replications are allocated among the evaluated points in the allocation step. Two iterations of the algorithm were executed here for illustration.

Figure 4.1 also shows the estimated global model with the initial design. Even though the estimation error of the global model is high, it correctly identifies the overall global trend of the model, indicating region 3 with the lowest trend. As the global observations of inducing points are not observable, we only indicate the location of inducing points in Figure 4.1. As shown, there are 3 local regions, each with two, one and two inducing points respectively. Based on the global model and inducing points, the gEI selects region 2 as the promising region with lower global trend and high uncertainty (smaller number of inducing points).

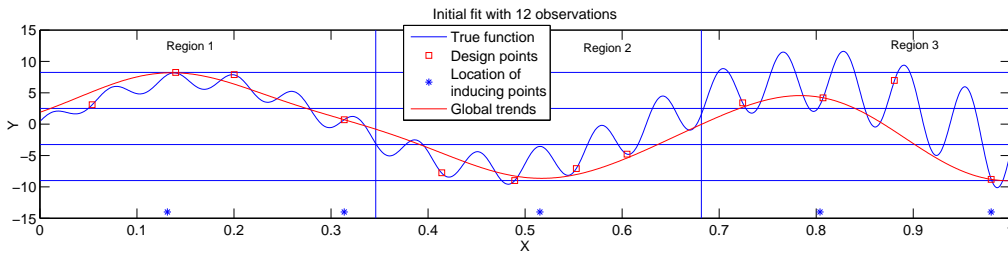


Figure 4.1: Initial fit

After the global search stage, the local search generates sets of candidate points and mEI is evaluated among them. In the first round of local search, the point with the highest mEI is evaluated with $r_{min} = 20$ replications. With the new evaluated point, the clusters and the inducing points are re-generated in region 2. The inducing points however do not change and the gEI does not significantly decrease. Hence, another

round of local search is conducted in this promising area with the newly generated local candidates, and mEI is evaluated. With little change in the inducing points and gEI , the local search is continued for another 3 rounds. After 4 new evaluated points are added in this local region 2, the algorithm stops the local search step and executes the allocation step where 10 additional replications are allocated to the evaluated points according to the criterion in Section 4.3.3.2 to reduce the stochastic noise in estimation and then return to the global search. We observe that the allocation step focused on placing more replications at points with high noise variance or near the contour lines.

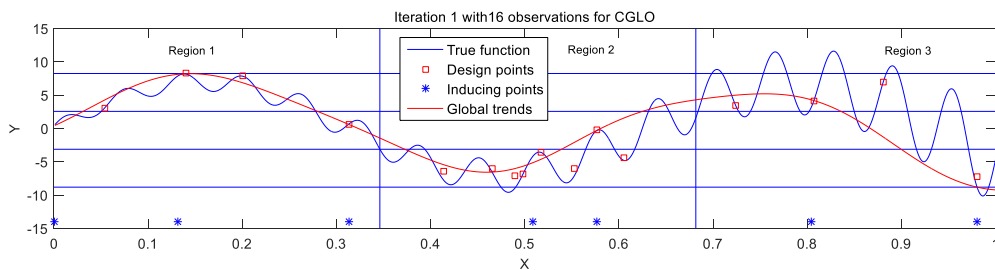


Figure 4.2: Iteration 1 of CGLO

After the first iteration, a total of 16 observations are obtained as seen in Figure 4.2. The global model is updated, and as seen, the global model is now better able to capture the overall trend in region 2, leaving regions 1 and 3 less explored. With region 3 having a lower trend between $[0.9, 1]$, the global search selects region 3 as the promising region in the next iteration. The local search is initialized at $x_g^* = 0.96$ with a low global trend and sparse distribution of observations around it. It follows the same local search as iteration 1 and adds 3 more observations (Figure 4.3).

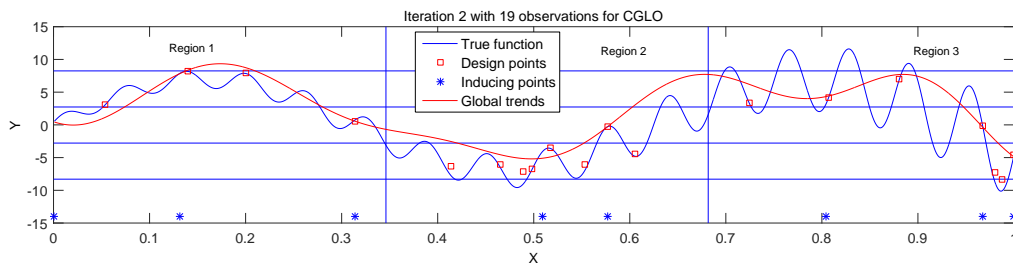


Figure 4.3: Iteration 2 of CGLO

4.5. NUMERICAL RESULTS

In the subsequent iterations, the search swings mainly between region 2 and region 3, and stops at region 3 with an estimated relative error (based on the estimated optimum and true optimum) of less than 1% after 9 iterations.

Overall, in this example, it is highlighted that the algorithm identifies promising regions efficiently by global search and it is also able to search more finely for optimal solution in that region by local search.

4.5.2 Comparative Studies with other Optimization Algorithms

In this section, we compared the CGLO algorithm with other surrogate-based optimization algorithms. We adopt the following example from Sun et al. (2014)

$$\max_{0 \leq x_1, x_2 \leq 100} g(x_1, x_2) = 10 \cdot \frac{\sin^6(0.05\pi x_1)}{2((x_1-90)/50)^2} + 10 \cdot \frac{\sin^6(0.05\pi x_2)}{2((x_2-90)/50)^2}. \quad (4.15)$$

g has a global optimal of $g(90, 90) = 20$ and the second best local optimal is $g(70, 90) = g(90, 70) = 18.95$ (see Figure 4.4). We introduce a noise term that is normally distributed with mean 0 and variance $\sigma_\epsilon^2(x_1, x_2) = 3(1 + x_1/100)^2(1 + x_2/100)^2$. With limited budget, this problem requires an efficient balance between global and local search because it has 25 local optimums and the difference between the largest and the second largest local optimal values is quite small. By introducing a large noise variance at the optimal area, we make the problem more challenging to solve.

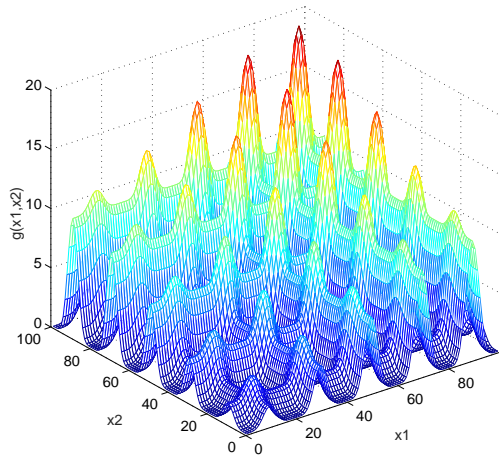


Figure 4.4: $g(x_1, x_2)$ function

Two alternative surrogate-based algorithms that apply the kriging metamodel, TSSO (Two Stage Sequential Optimization) (Quan et al., 2013) and EQI (Expected Quantile Improvement)(Picheny et al., 2013) are considered here. Both TSSO and EQI applied the MENK model as the surrogate, and were developed for optimization of simulations with heterogeneous noise. TSSO uses the OCBA allocation schemes and we select the online allocation scheme for EQI. We use a fixed wall clock time of 1400 seconds and fixed simulation replications of 5,000 and 10,000 for the comparison. In each comparison, all algorithms started off with the same 40 LHS design points with 20 replications at each design point. The number of replications for each new design point is chosen to be 10.

From Figure 4.5, we find that averaged over 30 macro-replications, the CGLO converges much faster to the optimal value than the TSSO and EQI. Although the CGLO performs poorer in the first few iterations (due to its approximation), its computational advantages and fast convergence are quickly apparent as the iteration goes on.

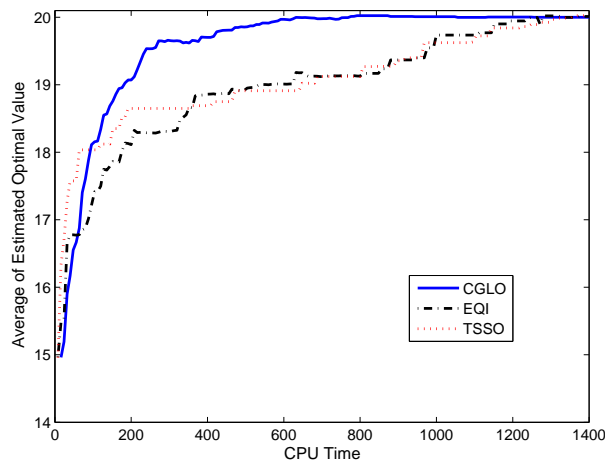


Figure 4.5: Estimated optimal value of TSSO, EQI and CGLO with CPU time of 1400s.

Even though the CGLO numerically converges faster than TSSO and EQI with limited time or number of replications, TSSO and EQI both catch up when more replications are used. Table 4.2 shows the average distance between the observed optimal solution and true optimal solution $|\Delta x|$ and the average distance to the true optimal value $|\Delta y|$ for each of the algorithms. As shown, CGLO performs significantly better than TSSO and EQI at $\alpha = 0.05$ in terms of optimal value and optimal solutions (denoted by $*$) when there are only 5000 replications, while there is no significant difference between the

4.6. A NAVIGATIONAL SAFETY PROBLEM

three algorithms when the simulation replication budget goes up to 10,000. The CGLO algorithm, even with the AGLGP model as an approximation, was able to achieve the level of accuracy similar to the TSSO and EQI in terms of optimal value and distance to optimal solution.

Table 4.2: Average performance with 5,000 and 10,000 simulation replications

Number of replications		5000		10,000	
Performance measure		$ \Delta x $	$ \Delta y $	$ \Delta x $	$ \Delta y $
CGLO	Average	0.6185*	0.2498*	0.5156	0.2097
	Standard deviation	2.6465	0.4263	0.2643	0.1928
TSSO	Average	12.5764	0.8746	0.5166	0.2106
	Standard deviation	13.3413	0.6880	0.2656	0.1931
EQI	Average	13.5894	1.1919	0.5158	0.2099
	Standard deviation	14.2587	0.8992	0.2635	0.1924

Note: * indicates that the value for the approach is statistically smaller than the other approaches at $\alpha = 0.05$ level.

4.6 A Navigational Safety Problem

We next test our CGLO algorithm on a maritime safety problem. We apply it on the agent based simulator developed for the Safe Sea Traffic Assistant (S^2TA) in (Pedrielli et al.). S^2TA applies a look-ahead approach with an agent based simulation model (ABM) to detect potential conflicts/collisions and derives optimal safe paths for vessels through heavy traffic regions. At any time point, it applies a 10 minute look ahead with its ABM in order to determine the safety of the current trajectory and to detect potential conflicts. If a potential conflict of high risk is detected on a pre-specified trajectory of a vessel from A to B (see Figure 4.6), an optimizer is then called to find an alternate trajectory that minimizes the probability of conflict within this period given the current and predicted traffic conditions.

Although S^2TA looks at several objectives (including the probability of conflict, deviation from original trajectory) while searching for an optimal alternative route, we

focus on the key safety objective of the probability of conflict in this example. As this system focuses on heavy traffic regions, this response can be highly non stationary (as a small turn can result in a very different conflict environment). Figure 4.7(a) shows this response on one of the trajectory parameters for a vessel with 90 vessels in its proximity. The variance of this response also differs quite a bit in the negative and positive polarity of the turn.

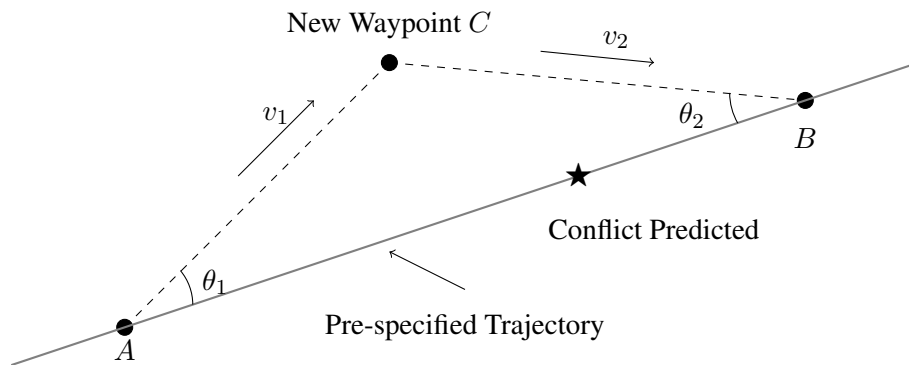


Figure 4.6: Definition of Trajectory

In this system, an alternative trajectory is then defined by its deviation from the pre-specified trajectory. Specifically, the alternative trajectory of a vessel is defined by a three point trajectory of $A - C - B$, which is characterized by the deviation angles θ_1, θ_2 and the travelling speeds on each leg $v_1(A - C)$ and $v_2(C - B)$ (see Figure 4.6). For practical reasons, the range for v_1, v_2 is between 1 and 15 [knots]. With positive and negative values on the angles indicating the deviations to the left or right from the original trajectory, θ_1, θ_2 are bounded between -60° to 60° , and both θ_1 and θ_2 are constrained on the same polarity.

Due to the stochastic nature in the movement of surrounding vessels, to evaluate the probability of conflict of a candidate alternate trajectory, M replications of the simulator are run, and the estimate is obtained over N additional replications. With the high nonstationarity of this response, the AGLGP model can be an appropriate surrogate for the response, and the CGLO algorithm will then be an effective one to locate an optimal alternate trajectory.

4.6. A NAVIGATIONAL SAFETY PROBLEM

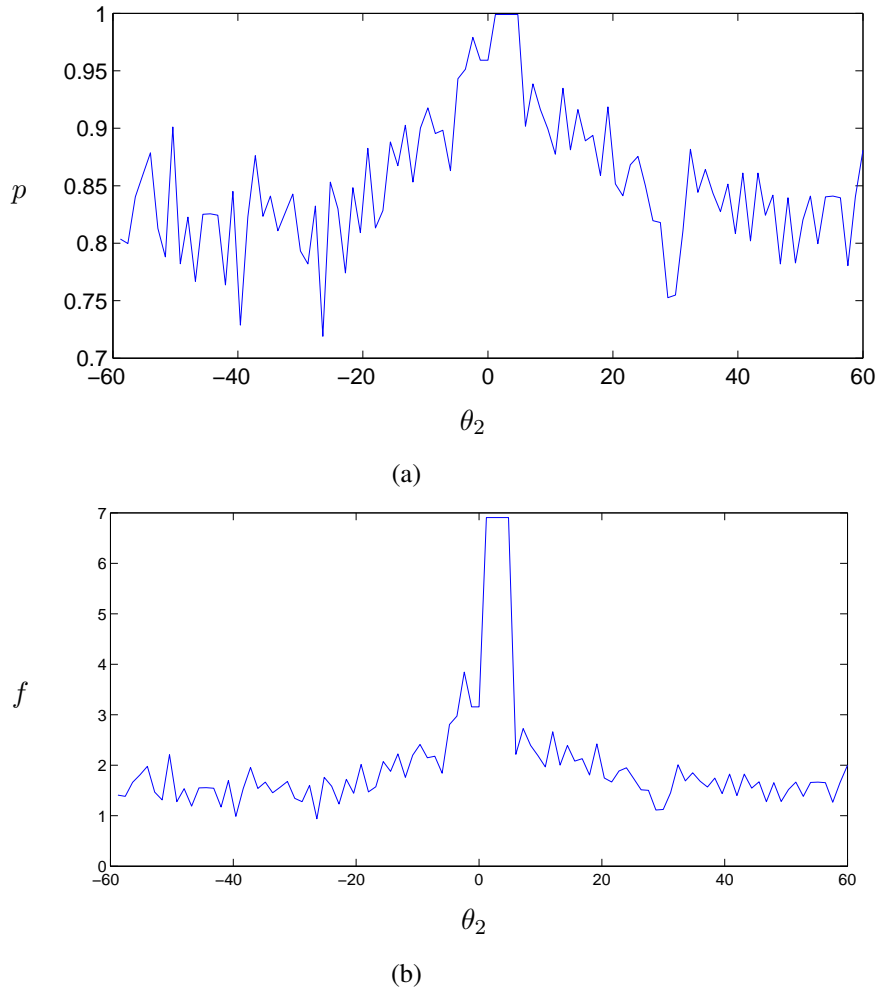


Figure 4.7: Probability of conflict in (a) and log transformation of probability in (b)

As this objective ranges in $[0,1]$, the following logistic transformation is applied, mapping the probability to the real line,

$$f(x) = -\ln\left(\frac{1}{p(x)} - 1\right) \quad (4.16)$$

while maintaining a similar variability and trend as the original probability estimate. With the transformation, the non stationarity of the response is still high (as seen in Figure 4.7(b)).

As the S^2TA is an on board real time system, determining an alternate trajectory needs to be fast when a conflict is detected. In this system, a 10 minute look ahead period is applied, and a maximum of 5 minutes is allowed to obtain an optimal solution. Fortunately, the simulator runs relatively fast and it takes 0.006 seconds to evaluate an

alternative trajectory. In this example, relatively many evaluations can be afforded, but may still be insufficient to comprehensively search the entire four dimensional space. Here we test CGLO with Random Search (RS) and TSSO.

We start the system with a crowded scenario based on historical data gathered from the Strait of Singapore. The system is run until a potential conflict is detected. At this point, we run CGLO, RS and TSSO to obtain the best optimal alternate trajectory. A maximum time for the search is set at 5 minutes before termination, at which point, the final best solution is determined. To evaluate the solution obtained from each approach, the 'optimal' solution was obtained from an extensive enumerative evaluation, which discretizes the solution space into 50×50 grids in both positive and negative polarity for θ_1 and θ_2 , and the speeds v_1 and v_2 in 15×15 grids and gives an estimation of minimum probability of $p^* = 3.6165 \times 10^{-4}$ at $x^* = [-57.6000, -52.8000, 5.4783, 11.7682]$. The 'optimal' solution was then compared with the solutions from the three approaches. Table 4.3 presents numbers of evaluation points and the distances between the 'optimal' solution y_{true} and the observed best solution by the each approach $y_{approach}$.

Table 4.3: The number of objective function evaluations and optimal probability of conflict $y_{approach}$ found by each optimization algorithm

Approach	Number of Evaluation Points	$\ y_{approach} - y_{true}\ $
RS	2000	0.0019
TSSO	256	0.0117
CGLO	611	9.4876×10^{-4}

As the simulation runs very fast, the random search can search at 2000 design points and identify the optimal area that is close to the true optimal location. Although none of the approaches managed to locate the optimal solution in the limited time, as seen from Table 4.3, both RS and CGLO perform quite well. CGLO is able to locate a safer solution (with a 50% lower probability of conflict) than RS.

Even though the TSSO works efficiently for optimization of expensive simulations, a large part of the time is spent on model estimation for this problem, and only 7,325 replications are simulated on 256 design points. With limited design points, the TSSO only identifies a local optimal solution at the positive polarity. With the fast approximated

4.7. CONCLUSIONS

AGLGP model and the combined searching structure, the CGLO can approximately afford for 16,264 replications on 611 design points and find the better optimal solution with a more extensive search.

A more extensive comparison is conducted to evaluate the solutions from 30 additional detected conflicts, and the averages obtained across the 30 conflict scenarios are shown in Table 4.4.

Table 4.4: The number of objective function evaluations and the deviation of the optimal probability of conflict $y_{approach}$ found by each optimization algorithm to the true optimal

Approach	Ave No. of Evaluation Points	Ave $\ y_{approach} - y_{true}\ $
RS	2000	5.76×10^{-3}
TSSO	311	0.0189
CGLO	625	1.58×10^{-3}

The random search shows the nearest average distance to the true optimal location with a more explorative search and a large number of evaluation points. The CGLO however finds a better probability of conflict with the global information of the response surface. The TSSO on the other hand fails to work efficiently for the problem with a fast simulation model. The results show that there is large potential for CGLO to be incorporated into the S^2TA system to provide much faster evaluations than TSSO (and hence, providing a more extensive search), and a better probability of conflict than RS. These benefits can be further amplified when the ABM in S^2TA is parallelized.

4.7 Conclusions

In this chapter, we proposed a combined global and local search algorithm based on the AGLGP model. The scheme systematically searches promising regions with a global stage and then searches more deeply into the region with the local stage. We then derived an allocation strategy that intelligently allocates budget to the evaluated points for the purpose of improving the metamodel fit and estimating the optimal solution. We analyzed its global convergence and studied its performance on a test function and a practical navigational safety problem. The results from this problem provided invaluable

insights on the application of the CGLO and indicated further extensions of CGLO in a parallel environment.

Chapter 5

PARALLEL GLOBAL AND LOCAL OPTIMIZATION WITH AGLGP MODEL

5.1 Introduction

As discussed in Chapter 4, the CGLO algorithm is a promising metamodel-based optimization algorithm. It has been shown to perform efficiently both with a limited number of simulation replications and with a limited CPU time. When the objective function has multiple local minima and can dramatically change over the design space, however, even CGLO may require a large number of function evaluations (possibly thousands of evaluation points) to get a reasonable solution. In such cases, CGLO can also become less efficient as the number of evaluation points increases in promising local regions. This situation becomes even worse for some real-time control systems which require effective response to any system changes over time.

In those real-time control systems whose behaviours change dynamically, a much faster optimization algorithm may be required because the systems need to effectively respond to those changes with new optimal settings in a short period of time. For example, in production scheduling, when a machine fails or a components' delivery delays, the system needs to respond effectively to these operational disruptions with an appropriate rescheduling policy to ensure that the products with right quantity or quality continue

to be produced. If the component processing time is in the order of an hour, a response has to be quickly made within 5 minutes. In maritime transportation, the Safe Sea Traffic Assistant (S^2TA) (Pedrielli et al.) applies a look-ahead approach with an agent based simulation model (ABM) to detect potential conflicts/collisions for vessels in heavy traffic regions. At any point, it looks 10 minutes ahead to determine the safety of the current trajectory and detect potential conflicts. If a potential conflict of high risk is detected on the pre-specified trajectory of a vessel, the optimizer is called. Then the optimizer needs to find an alternative trajectory that minimizes the probability of conflict within 5 minutes given the current and predicted traffic conditions.

To solve such problems where response time is critical, fast optimization algorithms like direct search methods can be applied as their speed is essential to conduct extensively fast search. On the other hand, the global and local information in CGLO can also be very useful and informative, especially for a large design space or highly complex functions. This is because they can help to drive the search to the correct optimal region for highly complex functions.

The speed of CGLO can be further improved by incorporating parallelization techniques. In recent years, parallelization techniques have been developed for many metamodel-based optimization methods. In particular, most existing parallelization approaches are able to deal with the generation of multiple distinct points in each iteration so that multiple simulation models can be evaluated simultaneously. Sóbester et al. (2004) essentially parallelized the EGO methods (Jones et al., 1998) by generating multiple evaluation points that have the best local maximums of the EI functions. Ginsbourger et al. (2008) first introduced the multivariate Expected Improvement (q-EI) and implemented it via Monte Carlo sampling. The implementation of q-EI is further studied by Clark and Frazier (2012) and Chevalier and Ginsbourger (2013). The multipoint sampling criteria for simultaneous evaluations have significantly reduced the computational time in design and analysis of computer experiments (Gramacy and Lee, 2009; Chevalier et al., 2014). In the literature of machine learning, the multivariate parallel optimization, also known as batch Bayesian optimization, has also been widely studied recently. For example, Desautels et al. (2012) and Contal et al. (2013) developed a parallel Gaussian Process algorithm based on the upper and lower confidence bounds.

The speed of CGLO can also be improved by incorporating some fast direct search methods. Many metamodels have been used fruitfully in conjunction with direct search methods for optimization problems with complex computer simulations. The asynchronous parallel pattern search (APPS) and the treed Gaussian Process (TGP) model are combined to generate a set of candidate locations that are queued for evaluations (Taddy et al., 2009; Gray et al., 2007). The mesh adaptive direct search (MADS) also uses the TGP as a surrogate and to evaluate the EI criterion (Gramacy and Digabel, 2015). Another common strategy for metamodel assisted direct search methods is to construct a coarse metamodel in the entire design space and use it to identify promising local regions. Then a more refined metamodel can be built in smaller local regions. It is then possible to explore several local regions simultaneously (Booker et al., 1999). The adaptive response surface method (ARSM) (Wang et al., 2001) disregarded regions with large function values as predicted by a surrogate and generated experimental designs using central composite designs (Montgomery, 1991) in the reduced region. The predictive uncertainty, however, is not well considered when reducing the design space based on the surrogate predictions. Hu et al. (2008) further extended the ARSM by using the particle swarm optimization (PSO) to refine the sampling in the reduced region. The application of PSO methods make it able to generate new samples according to the global and local history information. Sun et al. (2015) combined the global and local surrogates for fitness approximation in PSO. If any points in the population have better predictive fitness according to the surrogate, PSO will only evaluate at those points instead of evaluating the whole population. This approach, however, is still a meta-heuristic method that requires evaluation of the whole population when the surrogate fails to deliver a better predictive fitness among the population.

In this chapter, we propose a new parallel algorithm called parallel global and local optimization (PGLO) algorithm, which is a parallel implementation of CGLO. The general idea of PGLO is to globally search for multiple promising local regions and then locally generate a batch of evaluation points for simultaneous evaluations in each promising local region. It reduces both the computational time of the searching criterion by selecting multiple distinct evaluation points in each iteration and the computational time of running simulation models with simultaneous evaluations. Different from the

q-EI criterion in Ginsbourger et al. (2008), PGLO first searches globally for promising local regions with either lower global trend or few observations instead of searching for specific evaluation points. After identifying promising local regions to focus the search, the parallel local search is able to better exploit each promising local region for better solutions. Although there are different configurations of parallel computing environments, ranging from multiple-core personal computers or servers to clouds on the Internet, here we consider the multiple-core processors on personal computer or servers only as they are more readily available to general users. Hence it is assumed that the time of loading simulations to different processors and transmitting data among processors is almost negligible.

The chapter is organized as follows. In Section 5.2, we first discuss the desired properties of parallel search and evaluations. A simple one-dimension problem is then used to illustrate these desired properties. We review the background of the components in this approach in Section 5.3. The details of the parallel global and local search algorithm are provided in Section 5.4. Section 5.5 presents its convergence properties. Numerical results and conclusions are given in the following Sections 5.6 - 5.8.

5.2 Desired Properties of Parallel Search and Sampling Distributions

Although CGLO adopts the AGLGP model derived in Chapter 3 as a fast surrogate to provide efficient predictions with large data sets, for optimization problems where observations tend to be clustered in promising local regions, the local search can still become computationally challenging in regions with dense observations. Hence we expect a much faster optimization algorithm that can select multiple distinct evaluation points in each iteration for simultaneous evaluations without refitting of the AGLGP model after each evaluation. To achieve this, a parallel framework should at least have the following properties,

- evaluate multiple evaluation points quickly around the current best local optimal solution because it is likely to find better solutions around the current best one.

5.2. DESIRED PROPERTIES OF PARALLEL SEARCH AND SAMPLING DISTRIBUTIONS

- evaluate multiple evaluation points around different local optimal solutions because it better explores the whole space for a global optimal solution.

To better understand the desired properties of a parallel framework, we first look at the following example with the noisy test function

$$y(x) = (2x + 9.96)\cos(13x - 0.26) + \epsilon(x) \quad (5.1)$$

where $\epsilon(x)$ is the noise function that is normally distributed with mean 0 and variance $\sigma_\epsilon^2 = 4$ and $x \in [0, 1]$. The test function has a local minimum at 0.2628 and a global minimum at 0.7460. Here we adopt a single region for simple illustration starting from 7 Latin Hypercube Design (LHD) points and 10 replications at each point. The initial AGLGP model is given in the left plot of Figure 5.1. CGLO is applied and we see from the right plot of Figure 5.1 that the subsequent 6 more points selected by CGLO focuses on exploiting a local optimal area. The optimization procedure can be accelerated if there was a mechanism that could identify the local optimal area and quickly select multiple evaluation points around it. Direct search methods can be applied to serve this purpose. For example, by applying the pattern search method (Torczon, 1997) which selects new evaluation points quickly from a predefined lattice in a small neighbourhood, we can obtain a sequence of design points that focus on exploiting the same local optimal area as CGLO does (Figure 5.2).

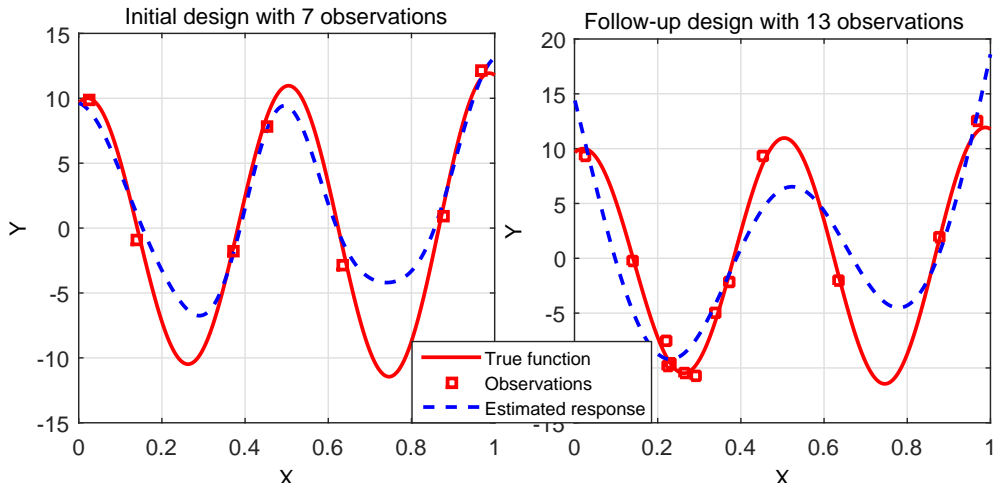


Figure 5.1: AGLGP model fit with design point selected by modified Expected Improvement

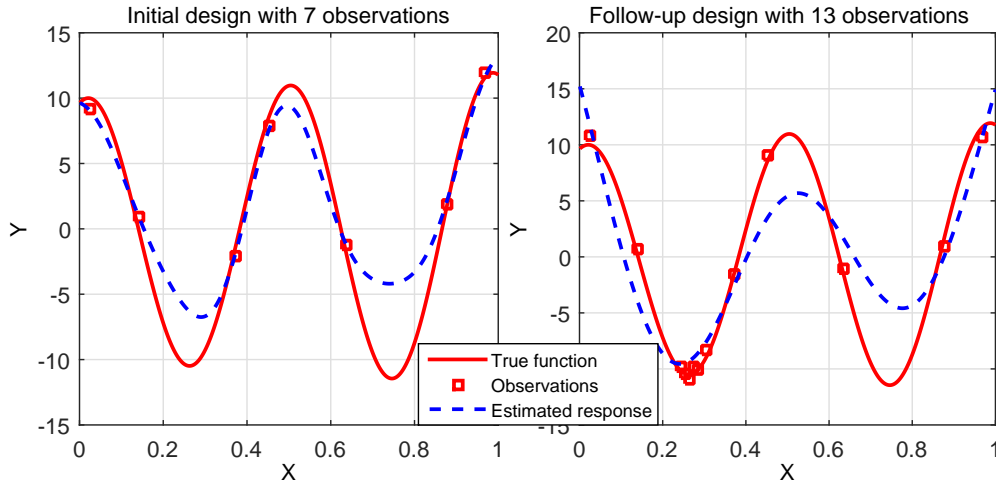


Figure 5.2: AGLGP model fit with design point selected by pattern search

Both CGLO and pattern search, however, have identified the wrong global optimal point in the first 6 iterations because of the low observations near the local minimum on the left. This problem could be mitigated if multiple local minimums can be identified and exploited simultaneously. Based on the same initial fit, the mEI function is displayed in the left panel in Figure 5.3. The two local maximums of the mEI function are selected for simultaneous evaluations, followed by pattern search exploiting around each of the two points. In this example, the two local maximums of the mEI function are located around the two local minimums of the function too. In general, this will not be the case. The maximum mEI locations can indicate areas with high spatial uncertainty but not low (or optimal) predictions. In these cases, however, pattern search can still guide the search towards local minimums because pattern search can continue exploiting for better solutions. Hence, the mEI and pattern search can combine to better exploit multiple local minimums simultaneously. As we can see in the right panel of Figure 5.3, the global optimal solution is much improved with two series of pattern search (searching for the two local minimums at the same time).

To get the desirable properties, here we propose the PGLO algorithm to consist of a global search stage to identify multiple promising local regions and a parallel local search stage searching for multiple local optimal solutions. The evaluations in the local search stage are initialized from a set of initial points that have either low predictions or high spatial uncertainty. In this way, multiple evaluation points from different potential

5.2. DESIRED PROPERTIES OF PARALLEL SEARCH AND SAMPLING DISTRIBUTIONS

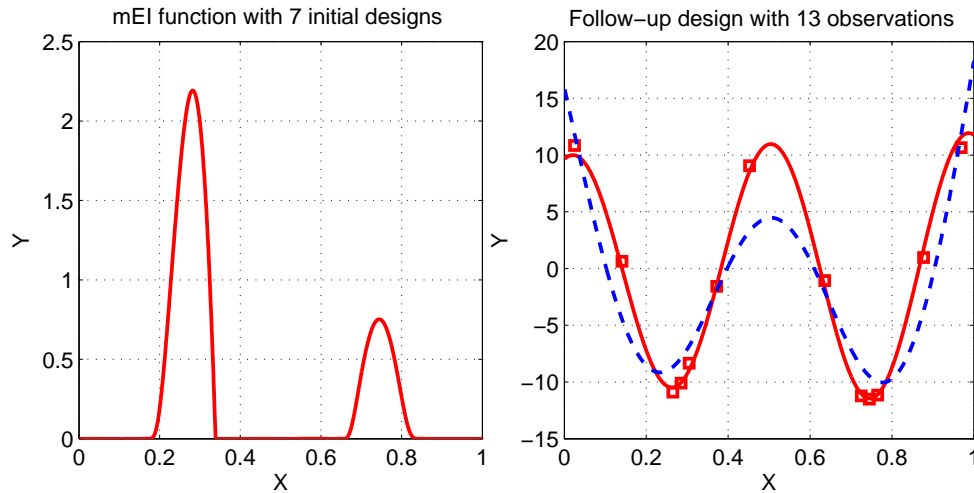


Figure 5.3: mEI function and AGLGP model fit with design point selected by multistart pattern search

local optimal areas are evaluated. Then, some direct search methods are applied to select additional follow-up evaluation points to quickly exploit around each of these initial points for better local optimal solutions. By doing so, we ensure that PGLO satisfies the two properties for a parallel framework as described at the beginning of this section. In Section 5.4, we explain the development of PGLO that satisfies the desirable properties in detail. To better understand each component in PGLO, a review of the relative criteria is first conducted in Section 5.3.

Selecting the best local maximums of the mEI function is a simple choice to select the initial evaluation points, and a more sophisticated selection criterion q-EI is introduced in Section 5.3.1. Moreover, pattern search is used as an example when deriving the PGLO algorithm. The choice of pattern search scheme is made on the grounds of its specific search patterns which are discussed in Section 5.3.2. It is noted that PGLO is not restricted to the pattern search algorithm, and in fact, any direct search method can be adopted. Within each promising local region, the *mEI* function based on the AGLGP model helps to explore the local region and direct search methods help to exploit for local optimal solutions.

5.3 Basics and Background: Multi-point Expected Improvement and Pattern Search

5.3.1 Multi-point Expected Improvement

The Multi-point Expected Improvement (also called qEI) criterion to find a q batch follow-up optimal design after n initial evaluated points was first defined in Schonlau et al. (1998), where the generalized q -point improvement is

$$I^g(x_{n+1}, \dots, x_{n+q}) = [\max(0, y_{min} - y(x_{n+1}), \dots, y_{min} - y(x_{n+q}))]^g \quad (5.2)$$

Here g is a user-defined tuning parameter where the larger the integer power g , the more globally the algorithm tends to search. Taking $g = 1$ as an example, by maximizing the expectation of the improvement $I(x_{n+1}, \dots, x_{n+q})$, we can deliver the follow-up designs in a batch single step as,

$$x_{n+1}^*, \dots, x_{n+q}^* = \arg \max_{\mathbf{x} \in \mathbb{X}^q} qEI(\mathbf{x}) = \arg \max_{\mathbf{x} \in \mathbb{X}^q} E[I(x_{n+1}, \dots, x_{n+q})] \quad (5.3)$$

The computation of qEI , however, becomes intensive as q increases. Moreover, the optimization problem in Equation (5.3) of dimension $q \times d$ requires demanding computational effort. Hence some approximations (via sequential evaluation) have been proposed. Sacks et al. (1989) first proposed the sequential approximation approach in the design of experiments that minimizes the mean square error. This approach sequentially designs one point at a time to reduce the computational burden from a single $q \times d$ dimensional optimization to a sequence of d -dimensional optimization. Although the effect of the sequential approach is difficult to be theoretically analyzed, this approach has been widely and successfully applied for many computer experiments (Santner et al., 2003; Jones et al., 1998).

Schonlau et al. (1998) sequentially optimized the q points one at a time. Once x_{n+1} is optimized, where $x_{n+1}^* = \arg \max E[\max(0, y_{min} - y(x_{n+1}))]$, x_{n+1}^* is assumed to be 'observed' when optimizing x_{n+2} . When optimizing the point x_{n+i} , the expected

improvement is

$$E(I(x_{n+i})) = E[\max(0, y_{min} - y(x_{n+i})) | x_{n+1}^*, \dots, x_{n+i-1}^*]. \quad (5.4)$$

where $y(x_{n+i}) \sim N(\hat{y}(x_{n+i}), \hat{s}_{n+i-1}^2(x_{n+i}))$. The predictive mean $\hat{y}(x_{n+i})$ is the Gaussian Process predictor (as shown in Equation (2.9)) given the initial n observations and does not change with new selected point $x_{n+1}^*, \dots, x_{n+i-1}^*$. The predictive variance $\hat{s}_{n+i-1}^2(x_{n+i})$, on the other hand, is the mean square error (as shown in Equation (2.10)) calculated based on all 'observed' points $x_1^*, \dots, x_{n+i-1}^*$.

Ginsbourger et al. (2010) proposed to update the expected improvement $E(I(x_{n+i}))$ sequentially with updated Gaussian Process model when optimizing the point x_{n+i} , which is given by

$$E(I(x_{n+i})) = E[\max(0, y_{min} - y(x_{n+i})) | y_{n+1}, \dots, y_{n+i-1}]. \quad (5.5)$$

where $y(x_{n+i}) \sim N(\hat{y}_{n+i-1}(x_{n+i}), \hat{s}_{n+i-1}^2(x_{n+i}))$. The predictive mean $\hat{y}_{n+i-1}(x_{n+i})$ is the updated Gaussian Process predictor (as shown in Equation (2.9)) given all 'observations' y_1, \dots, y_{n+i-1} . However as $x_{n+1}^*, \dots, x_{n+i-1}^*$ are not really observed, their 'observations' $y_{n+1}, \dots, y_{n+i-1}$ are unknown. Two heuristic strategies, *Kriging Believer* and *Constant Liar*, are proposed to replace the unknown $y_{n+1}, \dots, y_{n+i-1}$ by some deterministic values. *Kriging Believer* assumes the 'observations' are equal to the Gaussian Process predictors, i.e. $y_{n+1} = \hat{y}_n(x_{n+1}), \dots, y_{n+i-1} = \hat{y}_{n+i-2}(x_{n+i-1})$, while *Constant Liar* assumes they are all equal to a constant value, i.e. $y_{n+1} = L, \dots, y_{n+i-1} = L$. The constant value L can be set as equal to the minimum of the initial n observations. The smaller the L , the more exploitative the algorithm will be.

5.3.2 Pattern Search

In the selection of design points in each local region, pattern search algorithms, which have been shown to perform well in small areas (Torczon, 1997), can work computationally faster than metamodel-based search criteria. Pattern search algorithms are characterized by their meshes and polling conditions. A mesh is a lattice on which the search for an iterate is restricted. At each iteration k , three basic steps are executed:

1. Generate a set of trial points Q_k within a mesh M_k around the current best point x_k .
2. (a) Obtain a set of function evaluations F_k from computer models. If $\exists x_{k+1} \in M_k$ such that $f(x_{k+1}) < f(x_k)$, the search is successful.
(b) Else, polling conditions are applied to refine the mesh. Generally the mesh M_{k+1} is obtained by halving the mesh size, i.e. $M_{k+1} = M_k/2$. Repeat Step 1.
3. Update the best point x_{k+1} .

Essentially, pattern search performs the search using a predefined "pattern" of points that are independent of objective functions f and the design points that have been observed. Hence, the computational time for generating trial points is almost negligible and the majority of the computational time is spent on function evaluations. In this way, it efficiently exploits a small area for a local optimal solution with a shrinking mesh size, but it may not be able to sufficiently explore the entire space.

5.4 Development of Methodology

In this section, we present the framework for a master-worker parallel global and local optimization (PGLO) algorithm, which is a parallel version of the CGLO algorithm proposed in Chapter 4. It is an efficient algorithm that can quickly select multiple promising local regions and focus on searching for multiple local optimal solutions in each promising local region. Here, we assume one master processor and q available worker processors. The details of this algorithm are introduced in the following subsection.

5.4.1 General Framework of the PGLO Algorithm

As with any metamodel-based optimization methods, we begin by fitting the AGLGP model in Equation (3.7) from an initial set of points (typically from a space-filling experimental design like Latin Hypercube Design (LHD)). After the initial fit, each subsequent iteration of the algorithm is composed of a global search stage based on the global model (Equation (3.5)) that exploits the entire space globally for several

promising local regions, and a local search stage based on the local model (Equation (3.6)) that exploits within each promising local region simultaneously. Considering a total budget of T , in each iteration t , the budget exhausted is denoted as B_t , where $B_t = B_{t,s} + B_{t,a}$. The budget $B_{t,s}$ is for the parallel local search stage to find new design points (preferred as a multiplier of q) and the budget $B_{t,a}$ is for the allocation stage to allocate replications to existing design points. Here we simply assume a constant budget B_t over iterations. n_{max} is the maximum number of new evaluation points at each iteration. p_{max} is the maximum number of new evaluation points selected from pattern search before refitting of the local models. An overview of the algorithm is given in Table 5.1.

Specifically when $q = 1$, the global search stage in PGLO reduces to selecting only one candidate point that maximizes the gEI function (same as the global search stage in CGLO). The local region with the selected candidate point is then selected as the promising local region. Different from the local search stage in CGLO, PGLO accelerates the local search stage by incorporating pattern search as a fast alternative to the local search step. Initialized from the candidate that maximizes the mEI function based on the overall AGLGP model, the follow-up pattern search can either keep jumping towards a local optimal solution with a constant large mesh size or exploiting around a local optimal solution with a shrinking mesh size.

In the next subsection, we describe in detail Steps 3 and 4. Here, we fit the global model and conduct the global search on the master. Then the local models are fitted and used to select q distinct points for simultaneous evaluations on the workers. As the multi-point global search criterion is computed based on the fast estimated global model in Equation (3.19) with only a small number of inducing points and no simulation is run in the global search stage, the computational time for the global search stage on the master is much less than the local search stage on the workers.

5.4.2 Global Search Stage

In the global search stage in Step 3, PGLO focuses on identifying promising local regions to focus the search. It randomly generates a set of candidate points Ω_g and selects q points

Table 5.1: Overview of PGLO

Parallel Global and Local Optimization Algorithm	
<i>Step 1:</i>	<i>(Initialization)</i> Run a size n_0 space filling design, with r_{min} replications allocated to each point. Total initial replications $B_0 = n_0 r_{min}$. Set $t = 0$.
<i>Step 2:</i>	<i>(Validation of overall model)</i> Fit an AGLGP response model to the set of sample means and variances, and use cross validation to ensure that the AGLGP prediction is satisfactory.
	While the available replications $A = T - \sum_{i=0}^t B_i > 0$, $t = t + 1$
<i>Step 3:</i>	<i>(Global Search Stage)</i> Generate n_c candidate points Ω_g . Select q points x_1^g, \dots, x_q^g from Ω_g based on the global model and a multi-point global criterion. Identify the promising local regions, where x_1^g, \dots, x_q^g are located. Each promising local region \mathbf{D}_k with q_k selected candidate points is allocated with q_k processors for the local search and evaluation. Hence $q_1 + \dots + q_K = q$.
<i>Step 4</i>	<i>(Parallel Local Search Stage)</i> While $n_t < n_{max}$ and $A > 0$, <i>(Fit/Update local models)</i> Fit or update the local models in all the promising local regions. <i>(Generate Candidate Points)</i> Randomly generate n_l candidate points Ω_l^k independently in each promising local region \mathbf{D}_k . <i>(Select the Initial Evaluation Points)</i> In each promising local region \mathbf{D}_k , select q_k points $x_{n_t}^1, \dots, x_{n_t}^{q_k}$ from the candidate points Ω_l^k based on the overall model and a multi-point local search criterion. Simultaneously evaluate at all q selected points $x_{n_t}^1, \dots, x_{n_t}^q$ from all promising local regions with r_{min} replications on q processors. Hence the number of observed points $n_t = n_t + q$. <i>(Select the Follow-up Evaluation Points by Pattern Search)</i> Set $p_t = 0$, While $n_t < n_{max}$ and $p_t < p_{max}$, the q processors continue to evaluate at q new points from the predefined search patterns. Hence, $n_t = n_t + q$ and $p_t = p_t + q$. Break follow-up selection if a stopping criterion is satisfied. end end
<i>Step 5:</i>	<i>(Allocation Stage)</i> Allocate $B_{t,a}$ replications for additional evaluations among all evaluated points. end
<i>Step 6:</i>	<i>(Return the Optimal Solution)</i> Return the point with the lowest sample mean.

that maximize the multi-point global expected improvement from Ω_g . Each local region \mathbf{D}_k with q_k selected candidate points is then allocated with q_k processors for parallel local search. The generation of candidate points is similar to that in CGLO as introduced in Section 4.3.2.1. By smoothing out the localized features in each local region, the global search avoids putting efforts exploiting one single local region that has multiple neighboring local optimal solutions from the start.

5.4.2.1 Multi-point Global Expected Improvement

Here we apply a multi-point global expected improvement criterion as an extension of the gEI function in Equation (4.6). The q candidate points are selected through the maximization of the multi-point global expected improvement (q - gEI) function, which is defined as

$$q\text{-}gEI(x_1, \dots, x_q) = \mathbb{E}[\max\{(y_{gmin} - y_g(x_1))^+ \cdot \frac{1}{1 + e^{n_1/v-5}}, \dots, (y_{gmin} - y_g(x_q))^+ \cdot \frac{1}{1 + e^{n_q/v-5}}\}] \quad (5.6)$$

Similar to maximizing the qEI function in Equation (5.3), maximizing Equation (5.6) can be computationally challenging when q increases. To address this, we approximate the maximization of Equation (5.6) by selecting the q points x_1^g, \dots, x_q^g sequentially through optimizing the global expected improvement in Equation (4.6). As we expect the global expected improvement function to change with each new point $x_i^g, i = 1, \dots, q - 1$ added, to deliver a set of distinct points spread out across the entire space, we update the global model for each new x_i^g that has already been selected. To select x_{i+1}^g that optimizes the updated gEI function, the global model is updated given the new 'observation' $y(x_i^g)$ based on the *Kriging Believer* assumption, which assumes the observation value $y(x_i^g)$ equals to the overall AGLGP model prediction $\hat{y}(x_i^g)$ in Equation (3.7).

To ensure that q distinct candidate points are selected, if the global model does not change after each new x_i^g is added, a minimum number of *artificial points* \tilde{n}_i are assumed

to be selected around x_i^g by local search to affect the penalty term such that

$$gEI(x_i^g) = E[(y_{gmin} - y_g(x_i^g))^+] \cdot \frac{1}{1 + e^{(n_i + \bar{n}_i)/v - 5}} \leq \max_{x_c \in \Omega_g \setminus x_i^g} gEI(x_c), \quad (5.7)$$

This is a reasonable assumption because the local search stage will help to better exploit a promising area with a set of surrounding points around. This is also done as with a sparse distribution of the global candidate points, each new added point x_{i+1}^g with maximum gEI can indicate a different local region from the previous added point x_i^g . This avoids putting all effort in a single local region and allows the algorithm to spread the searching effort in multiple local regions.

5.4.3 Parallel Local Search Stage

Next we provide the details on the parallel local search stage. Once the promising local regions are selected by the global search, the local search stage then searches more extensively within each promising local region for better solutions. In this stage we first adopt a multi-point modified expected improvement function to select an initial set of q evaluation points for simultaneous evaluations, with q_k points selected from each promising local region \mathbf{D}_k . A follow-up pattern search is then conducted for additional evaluation points exploiting around each of the initial q points.

5.4.3.1 Selection of Initial Evaluation Points

The role of the initial points is to start the local search from promising areas, which have either low predictions or higher spatial uncertainty, and quickly progress to better solutions in the local region with additional follow-up evaluation points. In order to achieve this, we propose a multi-point modified expected improvement function as an extension of the mEI criterion in Equation (4.4) (Quan et al., 2013). The initial q_k evaluation points in the promising local region \mathbf{D}_k are selected by maximizing the q_k - mEI function defined as

$$q_k\text{-}mEI(x_1, \dots, x_{q_k}) = \mathbb{E}[\max\{(y_{min} - z(x_1))^+, \dots, (y_{min} - z(x_{q_k}))^+\}] \quad (5.8)$$

where y_{min} is the predicted response at the sampled points in the local region \mathbf{D}_k with the lowest sample mean, and $z(x)$ is a normal random variable with mean given by the AGLGP predictor in Equation (3.7) and variance given by spatial prediction uncertainty $\hat{s}_z^2(x) = L_{nn} - \mathbf{1}'\mathbf{L}_n^{-1}\mathbf{1}$. Instead of searching the global optimal solution for the batch (x_1, \dots, x_{q_k}) , we approximate it again by optimizing the q_k points sequentially one at a time with each step maximizing the $mEI(x)$,

$$\begin{aligned} x_1^* &= \arg \max_{x \in \Omega_l} mEI(x) \\ &= \arg \max_{x \in \Omega_l} E(\max[y_{min} - z(x), 0]), \end{aligned} \quad (5.9)$$

$$\begin{aligned} x_{i+1}^* &= \arg \max_{x \in \Omega_l} mEI(x) \\ &= \arg \max_{x \in \Omega_l} E(\max[y_{min} - z(x), 0] | x_1^*, \dots, x_i^*, y_1, \dots, y_i). \end{aligned} \quad (5.10)$$

Here we also optimize the mEI with respect to a set of candidate points Ω_l , which are uniformly distributed in the local region (Regis and Shoemaker, 2007). To select the optimal point x_{i+1}^* , the local model and the mEI function are updated with the selected points x_1^*, \dots, x_i^* and their 'observations' y_1, \dots, y_i . We approximate the 'observations' y_1, \dots, y_i with the AGLGP model predictions $\hat{y}(x_1), \dots, \hat{y}(x_i)$ in Equation (3.7). As the updated model variance $\hat{s}_z^2(x) = L_{nn} - \mathbf{1}'\mathbf{L}_{n+1}^{-1}\mathbf{1}$ is reduced around the selected points x_1^*, \dots, x_i^* , it avoids selecting new points near the selected ones.

5.4.3.2 Selection of Follow-up Evaluation Points

In this step, we continue the local search by selecting additional follow-up evaluation points to quickly exploit the promising areas around each of the q initial points for better solutions in the promising local region. To achieve this, we apply q pattern search to quickly select q new evaluation points from the pre-defined search patterns. The pattern search is initialized from each of the q initial points obtained as described in Section 5.4.3.1, and it is sequentially repeated with q new selected evaluation points. Although the pattern search can move efficiently towards a local optimal solution in a small area, it only converges at a local optimal solution. As there may exist more than one local optimal solution in each local region, it is not desirable for the algorithm to converge only to a local optimal solution instead of a global optimal solution in the local region.

Hence, to avoid being trapped in a local optimal solution, we stop the pattern search when a stopping criterion is satisfied and require the algorithm to reselect a new set of initial points that explore the entire local region for potentially better local optimal solutions.

Stopping Criterion In the follow-up pattern search step, the mesh size of the pattern search can keep shrinking as the search progresses and it already identifies a local optimal point, $M_{t_k+1} = 1/2M_{t_k}$. To avoid spending unnecessary budget for unsuccessful searches, we set a stopping criterion where we stop the pattern search when

$$M_{t_k} \leq M_{min} \quad (5.11)$$

where M_{t_k} is the current mesh size and M_{min} is a predefined minimum mesh size (Torczon, 1997). When one of the q pattern search stops at a local minimum before p_{max} budget is exhausted in that promising local region \mathbf{D}_k , we stop the follow-up points selection step and require the algorithm to return to the initial points selection step to select a new set of initial points. The local models are then updated in the promising local regions, and new initial evaluation points are selected by maximizing the updated mEI function to identify new promising areas.

This stopping criterion allows the pattern search to escape from the current local optimal solution to a potentially better local optimal solution before exhausting all the p_{max} budget. The new initial points can be selected with either low prediction or high predicted mean square error. If the new initial points have low predictions, the pattern search continues exploiting the current promising areas, and if the new initial points have high predicted mean square error, the pattern search starts to exploit new promising areas.

5.4.4 Allocation Stage

In this stage, we adopt an allocation strategy to evaluate the best few optimal solutions after the local search with an additional number of replications $B_{t,a}$. The approach adopted here is similar to the allocation procedure in CGLO. Specifically, the Optimal Computing Budget Allocation (OCBA) approach (Chen and Lee, 2010) is adopted for all

the already evaluated n points. Suppose each point x_i has a sample mean given by \bar{y}_i and a sample variance $\sigma_\epsilon^2(x_i)$. Then according to Theorem 3.2 provided by Chen and Lee (2010), the Approximate Probability of Correct Selection (APCS) can be asymptotically maximized (as the available budget tends to infinity) when

$$\frac{N_{i,b}}{N_{j,b}} = \left(\frac{\sigma_\epsilon(x_i)/\Delta_{b,i}}{\sigma_\epsilon(x_j)/\Delta_{b,j}} \right)^2, i, j \in \{1, 2, \dots, n\}, i \neq j \neq b \quad (5.12)$$

$$N_{b,b} = \sigma_\epsilon(x_b) \sqrt{\sum_{i=1, i \neq b}^n \left(\frac{N_{i,b}}{\sigma_\epsilon(x_i)} \right)^2}$$

where \bar{y}_b is the lowest observed sample mean in the entire space, $N_{i,b}$ is the number of simulations allocated to point x_i , and $N_{b,b}$ is the number of simulations allocated to the point x_b with the lowest sample mean. Hence, $\sum_i N_{i,b} = B_{t,a}$. $\Delta_{b,i} = \bar{y}_i - \bar{y}_b$. The OCBA technique is able to allocate additional replications to the points with low sample means and high sample variances in order to distinguish the best point from the other competitors. With additional replications allocated at those potential optimal points, the model estimation around those points can be further improved, which in turn can help the global and local search criteria to make better selection in the subsequent iteration. After the allocation stage, the sampled point with the lowest sample mean is selected as the location of the current best response.

5.5 Convergence of the PGLO Algorithm

Under some mild conditions, the above PGLO framework can be shown to be convergent in a probabilistic sense using a similar argument to the one in CGLO. We first show that when there is only one processor, PGLO will converge to the unique global optimal solution f^* at the global optimizer x^* . Following the same arguments from CGLO, to show this here, we need to show that the Lemma 4.2 in Chapter 4 also holds true under the framework of PGLO.

Lemma 5.1. *If any $x \in \Omega_g$ is selected infinitely often (i.o.) by global search, the set of points observed \mathbb{X}_t is dense in \mathbb{X}_Ω as $t \rightarrow \infty$*

Proof. To prove that the set of points observed \mathbb{X}_t is dense in \mathbb{X}_Ω , it suffices to prove that each local region \mathbf{D}_k is dense as $\cup_k \mathbf{D}_k = \mathbb{X}_\Omega$. Based on Lemma 4.1, each global candidate is selected infinitely often, hence each local region is allocated with infinite budget.

As we follow the SRS framework (Regis and Shoemaker, 2007) in the local search stage, which generates candidate points from a uniform distribution in each local region, from Theorem 1 in Regis and Shoemaker (2007), the sequence of observed global optimizer will converge to the true global optimizer $x_t^* \rightarrow x^*$ almost surely. Hence the observed global optimizer in each local region \mathbf{D}_k will converge to the true global optimizer in \mathbf{D}_k . Furthermore, from Theorem 1.3 in Torn and Zilinskas (1989) that is restated below, we have dense local region \mathbf{D}_k for $\forall k$.

Theorem. 1.3 (Torn and Zilinskas, 1989) *Let the minimization region A be compact. Then a global minimization algorithm converges to the global minimum of any continuous function iff the sequence of trial points of the algorithm is everywhere dense in A .*

□

Hence, following Lemma 4.3 and Theorem 4.4, we have that the observed global optimizer in the entire space will converge to the true global optimizer $x_t^* \rightarrow x^*$ and the observed global optimal solution will converge to the true global optimal solution $\bar{y}_t^* \rightarrow f^*$ w.p.1 as $t \rightarrow \infty$. Then for any $q > 1$, we show that the points by local search get dense in the entire space.

Lemma 5.2. *Suppose the set of points observed on each processor p_i until iteration t is $\mathbb{X}_t^{p_i}$, $\cup_q \mathbb{X}_t^{p_i}$ is dense in \mathbb{X}_Ω as $t \rightarrow \infty$.*

Proof. It is sufficient to show that the first processor delivers dense observations. In both global and local search stage, we approximately optimize the multi-point selection criteria by sequential optimization approach.

$$x_1^g = \arg_{x \in \Omega_g} \max gEI(x) = \arg_{x \in \Omega_g} \max E\{\max(y_{gmin} - y_g(x), 0)\} \cdot \frac{1}{1 + e^{n_i/v-5}}$$

The local region which contains x_1^g is selected as the promising local region, and the first processor will be allocated to evaluate at the point x_1^* that maximizes the mEI function in this local region, where

$$x_1^* = \arg \max_{x \in \Omega_l} mEI(x) = \arg \max_{x \in \Omega_l} E(\max[y_{min} - z(x), 0]). \quad (5.13)$$

Given Lemma 4.1 and Lemma 5.1, this processor will generate dense evaluations, and $\cup_q \mathbb{X}_t^{P_i}$ is dense in \mathbb{X}_Ω as $t \rightarrow \infty$. \square

Hence, following the same argument in Theorem 4.4, for $q > 1$, PGLO also converges, i.e. $\bar{y}_t^* \rightarrow f^*$ and $x_t^* \rightarrow x^*$ w.p.1 as $t \rightarrow \infty$.

5.6 Numerical Studies

In this section, we first numerically evaluate the effect of including a pattern search in PGLO compared with CGLO. Then we compare the performance of the proposed PGLO with other parallelized pattern search techniques on different test functions.

5.6.1 Comparison with CGLO

We consider a simple one-dimensional test example $\cos(100(x - 0.2)) \exp(2x) + 7 \sin(10x)$ with the random noise $\epsilon(x) \sim N(0, 0.2 + 0.1 \sin(10x))$ to evaluate the efficiency of the incorporated pattern search in PGLO. The performance of PGLO with one processor $q = 1$ is compared with CGLO, which is based on a sequential local model-based search criterion in local search stage.

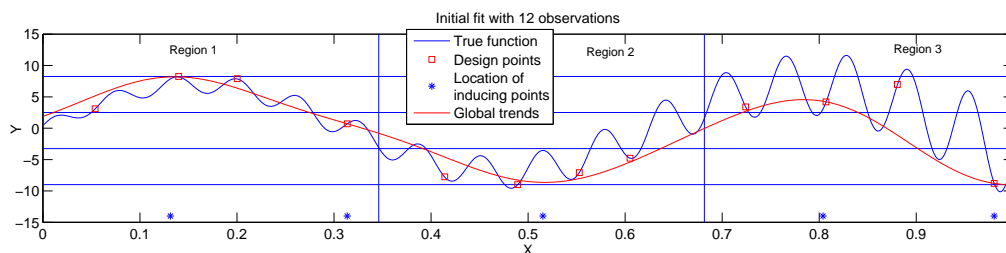


Figure 5.4: Initial fit

We initialize 12 design points with Latin Hypercube designs and 3 local regions, as seen in Figure 5.4. As shown, the global optimal solution is -10.1316 at $x = 0.9865$

and another sub-optimal solution is -9.5799 at $x = 0.4826$. Hence, regions 2 and 3 are more promising local regions with lower response that should be focused on. Each design point is evaluated with a minimum of 20 replications. 10 additional replications are allocated among the evaluated points in the allocation stage. Two iterations of each algorithm were executed here for illustration.

Both CGLO and PGLO apply the same global search criterion gEI . After the global search stage, they both identify region 2 as the promising local region. In the local search stage of CGLO, four new evaluated points were added in this local region 2 before it stopped the local search step and executed the allocation step. To fairly compare the distribution of the new design points, we sequentially selected four points from pattern search in the local search stage of PGLO.

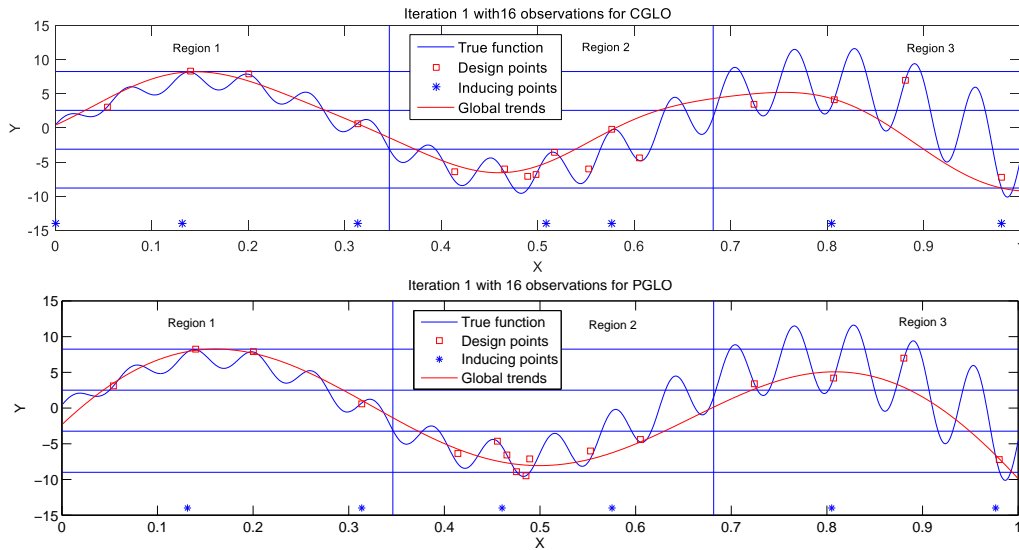


Figure 5.5: Iteration 1 for CGLO and PGLO

After the first iteration, a total of 16 observations are obtained as shown in Figure 5.5. As seen, CGLO was able to explore several local optimal areas in region 2, while PGLO with pattern search better exploited one promising area in region 2. Here, both CGLO and PGLO managed to search region 2 well and obtained a better solution. The global model was then updated and we see that it is now better able to capture the overall trend in region 2, leaving regions 1 and 3 less explored. In the second iteration, the global search stage in both algorithms was able to identify region 3 as promising and initialized the local search from $x_1^* = 1.000$. They both exploited the most promising area in region

5.6. NUMERICAL STUDIES

3 with additional three design points. With these three additional points, PGLO is able to find a better optimal solution (Figure 5.6).

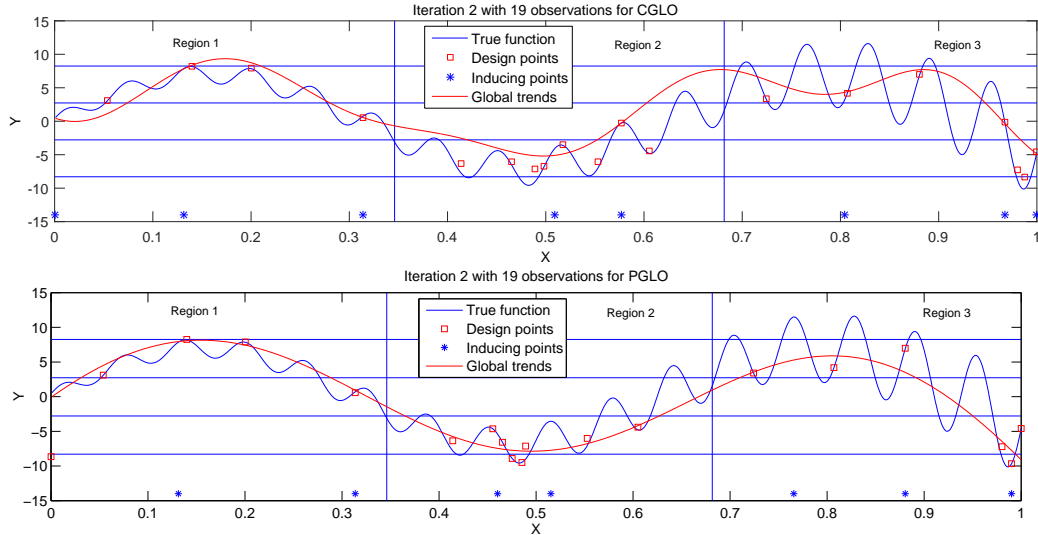


Figure 5.6: Iteration 2 for CGLO and PGLO

From Table 5.2, we can see that PGLO is able to deliver a smaller difference Δy between the observed optimal value and the true optimal value. This is because PGLO is better able to exploit each promising local area with the pattern search scheme. Although the optimal solution by PGLO has a larger distance Δx to the true optimal solution in iteration 1, CGLO fails to find a better solution in this iteration. An additional benefit of PGLO is its computational efficiency. The computational time for the predefined search patterns in the local search stage of PGLO is almost negligible while CGLO requires some additional time for refitting of the local models in the local search stage. This effect becomes more significant as the algorithm progresses and the number of evaluations observed gets large.

Table 5.2: Performance of CGLO and PGLO per iteration

Algorithm	Initial		Iter 1		Iter 2	
	Δy	Δx	Δy	Δx	Δy	Δx
CGLO	2.9118	0.0062	2.9118	0.0062	1.8344	0.0032
PGLO	2.9118	0.0062	0.6402*	0.5011	0.4582*	0.0028

Overall in this example, it is highlighted that both CGLO and PGLO identify the promising local regions efficiently with the global search. They are also able to search

more finely for optimal solutions in promising local regions with the local search. The local search in CGLO automatically balances the exploration and exploitation in a promising local region with the mEI searching criterion. PGLO, on the other hand, better exploits the specific promising areas for a better solution after it explores the whole promising local region. Even though the local pattern search is a locally convergent algorithm, PGLO avoids wasting budget exploiting a sub-optimal area by defining an intelligent escape policy. By incorporating a direct search method, PGLO works more efficiently computationally, especially for real-time control systems.

5.6.2 Comparison with Other Parallel Pattern Search Techniques

In this section, we compare the performance of PGLO (which incorporates the AGLGP model and the efficient AGLGP model-based global and local searching criteria to drive the local pattern search) with other parallelized pattern search techniques.

Here, we compare MultPPS-LHS, using a direct application of multistart parallel pattern search (*MultPPS*) initialized with Latin Hypercube Sample (LHS), and MultPPS-qEI, with *MultPPS* initialized from the multi-point expected improvement (q-EI) of the AGLGP model. MultPPS-LHS is used as a benchmark algorithm for the direct application of pattern search method, while MultPPS-qEI is compared to evaluate the performance of PGLO which incorporates pattern search within a global and local search algorithm against a single level search with q-EI. To replicate the situation of a fast simulation model, a wait time of 0.01 seconds is added to each function evaluation. The sequential pattern search is limited with 200 evaluations to avoid spending too much budget in any sub-optimal areas.

The performance of a parallel optimization algorithm is measured by the wall clock time consumed to find a reasonable solution within a specific level of accuracy. With known global optimum f^* , the relative error of a reasonable solution f_{best} , $|f^* - f_{best}|/|f^*|$, should be less than 1%. Another commonly used measure for the parallel performance is the *speedup*, which is defined by the time required for the sequential optimization on one processor $T(1)$ divided by the time required for the parallel optimization on q processors $T(q)$, i.e., $SP = T(1)/T(q)$.

5.6. NUMERICAL STUDIES

First we adopt the same example in equation (4.15) (Figure 4.4). Table 5.3 presents the average wall clock time to get a solution with a relative error $|g^* - g_{best}|/|g^*| < 1\%$. The observed global optimum g_{best} should at least be able to find the global optimal area with a relative error of less than 1%. Even though the simulation runs as fast as 0.01s, the direct application of pattern search (MultPPS-LHS) performs least efficiently for any number of processors. Because the objective function has 25 local optimal solutions, the MultPPS-LHS cannot explore the entire space sufficiently without global information. However, as the number of processors increases, the MultPPS-LHS catches up with better exploration.

Table 5.3: Average wall clock time to get a reasonable solution with a relative error $< 1\%$ using $q=1,4,8$ Processors

q	PGLO	MultPPS-LHS	MultPPS-qEI
1	210.3611	272.1705	252.2541
4	42.4318	92.1054	53.1587
8	37.0719	47.9258	39.6525

Note: This table shows, for each optimization algorithm, the average wall clock time required for a reasonable solution to be obtained from a sample of 30 macro-replications.

In this example, MultPPS-qEI performed significantly worse with one and four processors at $\alpha = 0.05$, but there is no significant difference between PGLO and MultPPS-qEI with 8 processors. This is because PGLO better explores the entire space in the first several iterations with a global search stage. Instead of identifying one particular local optimal area, the global search identifies promising local regions (which can include multiple local optimal areas). Once a point is picked up in one local region, it reduces the tendency to pick too many points in the same local region. Therefore, it balances between exploiting too much in the local region and spreading out more points for exploration in more local regions. When there are eight processors, MultPPS-qEI also has sufficient budget to explore each local region, and hence the difference becomes not significant.

Table 5.4: Relative speedup of parallel optimization algorithm when using $q=4,8$

q	PGLO	MultPPS-LHS	MultPPS-qEI
4	4.9576	2.9565	4.7547
8	5.6790	5.7872	6.4605

Table 5.4 presents the relative *speedup* when $q = 4$ and $q = 8$ to evaluate the efficiency of parallelization. The relative speedup measures how well a parallel algorithm scales relatively to its serial version with additional processors. It is worth mentioning that as different algorithms can require different wall clock times with one processor, a larger speedup does not mean a better algorithm in absolute time. It is also problem dependent. The results show that in this example, both PGLO and MultPPS-qEI have significant speedup for four processors, but only achieve marginal improvement with additional four processors (eight in total). This is because the additional processors will end up with exploiting the same optimal area. Even though they are initialized from different locations, different pattern search will deliver the same optimal solution. MultPPS-LHS, on the other hand, achieves significant speedup with any additional processors because they will help to better explore the entire space.

A more extensive comparison is conducted on four different two-dimensional multimodal test functions (Surjanovic and Bingham) to evaluate the performance of the parallel algorithms (see in Appendix E). The first three test functions, Griewank, Ackley and Levy, consist of large global variability and small local variability. The Griewank function has many widespread local minima, which are regularly distributed, based on the global trend. Ackley is characterized by a nearly flat outer region with small local variability, and a large hole at the center. Levy function changes dramatically with respect to one variable but relative stable with the other. The Schwefel function is on the other hand more complex with random bumpiness across the region. The noise variance function differs for each function based on the scale of their variability. Here we consider two levels of noise variance, the *small* noise variance with a maximum of 1% of the functional variability and the *large* noise variance with a maximum of 10% of the functional variability (i.e. range of the function values).

5.6. NUMERICAL STUDIES

Table 5.5: Average wall clock (W.C.) time and relative speedup to get a reasonable solution with a relative error $< 1\%$ using $q=1,4,8$ Processors (*small noise*)

Test	q	PGLO		MultPPS-LHS		MultPPS-qEI	
		W.C. Time	speedup	W.C. Time	speedup	W.C. Time	speedup
Griwank	1	23.2517	-	36.6972	-	27.4897	-
	4	11.2042	2.0753	30.0700	1.2204	11.6401	2.3606
	8	10.4059	2.2345	28.3077	1.2964	10.5410	2.6079
Ackley	1	21.0555	-	58.9847	-	31.9155	-
	4	12.8447	1.6392	25.6174	2.3059	16.2700	1.9616
	8	11.8217	1.7811	21.5801	2.7333	12.2501	2.6053
Levy	1	28.4872	-	62.5031	-	37.2374	-
	4	15.5415	1.8330	37.0023	1.6892	18.6369	1.9986
	8	15.3431	1.8567	31.9594	1.9557	15.7109	2.3700
Schwefel	1	56.6273	-	178.3245	-	68.2456	-
	4	23.6495	2.3941	57.8724	3.0814	30.2457	2.2564
	8	18.6162	3.0414	32.8356	5.4308	22.1437	3.0819

Note: This table shows, for each optimization algorithm, the average wall clock time required for a reasonable solution to be obtained from a sample of 30 macro-replications.

Table 5.5 shows the average wall clock time for each test function with small noise variance over 30 macro-replications. Although the wall clock time of PGLO is at least as less as the other algorithms in all scenarios, its relative speedup is not very satisfactory on the first three test problems. In addition, the relative speedup for all these parallel algorithms is generally poor on the first three test functions. This is because the first three test functions have large global variability and their global trends have only one global optimum that is significantly smaller than other local optima. Hence, by smoothing out the small local variability with a global model, the global optimal area is easy to identify and the addition of more processors will generally not make the search for the global minimum faster on such problems. With such functions, the MultPPS-qEI can be as competitive as PGLO in both wall clock time and speedup because they both focus their computing effort in one promising local region. The last test function, which has large

Table 5.6: Average wall clock (W.C.) time and relative speedup to get a reasonable solution with a relative error $< 1\%$ using $q=1,4,8$ Processors (*large* noise)

Test	q	PGLO		MultPPS-LHS		MultPPS-qEI	
		W.C. Time	speedup	W.C. Time	speedup	W.C. Time	speedup
Griwank	1	33.1786	-	36.6972	-	32.9710	-
	4	17.5062	1.8952	30.0700	1.2204	23.3297	1.4133
	8	17.1770	1.9316	28.3077	1.2964	18.6146	1.7712
Ackley	1	54.7717	-	128.2356	-	59.8706	-
	4	21.4285	2.5561	66.9703	2.3059	21.3066	2.8100
	8	16.8121	3.2579	52.3458	2.7333	18.4025	3.2534
Levy	1	72.5785	-	150.6540	-	53.3854	-
	4	23.9806	3.0266	79.5076	2.9491	33.6834	1.5849
	8	14.5598	4.9849	53.3341	3.8716	25.8670	2.0638
Schwefel	1	110.8477	-	298.3565	-	148.7304	-
	4	35.3226	3.1382	105.2468	2.8200	35.8572	4.1479
	8	23.5690	4.7031	48.7986	5.3806	26.9884	5.5109

Note: This table shows, for each optimization algorithm, the average wall clock time required for a reasonable solution to be obtained from a sample of 30 macro-replications.

local variability that dramatically changes over the entire space, shows similar results as the first demonstration example in Equation (4.15) (Figure 4.4) that PGLO performs significantly better than the rest.

Table 5.6 shows the computational performance for all these test functions with large noise variance. As shown, the wall clock time is generally larger than the examples with small noise. Additional replications are required for searching and evaluations with the highly noisy response surface. However, the relative speedup for all test functions with large noise is generally better than with small noise. This is because the availability of multiple evaluation points and the additional evaluations improve the accuracy of the AGLGP model in each iteration, which is helpful in determining the location of the global optimal solution. Overall from these test functions, the performance of PGLO is generally better than the MultPPS-LHS and at least as well as MultPPS-qEI. This

indicates that incorporating the pattern search with a global and local metamodel structure in PGLO make it more efficient than a simple parallelized pattern search and an only global metamodel driven pattern search when the objective functions exhibit high nonstationarity.

5.7 A Navigational Safety Problem

We next test our PGLO algorithm on the same maritime safety problem in Chapter 4 to find an optimal trajectory with the minimum probability of conflict. In this system, an alternative trajectory is defined by its deviation from the pre-specified trajectory, which is characterized by the deviation angles $\theta_1, \theta_2 \in [-60^\circ, 60^\circ]$ and the travelling speeds on each leg $v_1(A - C) \in [0, 15]$ and $v_2(C - B) \in [0, 15]$ (see Figure 4.6). Hence the optimal trajectory is searched in the large four dimensional solution space. As this system focuses on heavy traffic regions, this probability of conflict can be highly non stationary (as seen in Figure 4.7).

In this example, the simulator runs as fast as 0.006 second to evaluate an alternative trajectory and a maximum of 5 minutes is allowed to obtain an optimal solution. Even though thousands of evaluations can be afforded in this example, this budget may still be insufficient to comprehensively search the entire four dimensional space. As shown in Chapter 4, the CGLO can work better than the random search and the TSSO (Quan et al., 2013) with a smaller probability of conflict. However, none of the algorithms is able to identify the global optimal solution within the limited computational time. Hence, a much faster and effective searching procedure to find the global optimal solution is required.

PGLO has the desirable characteristics to quickly and more effectively search the entire space. With the fast estimated AGLGP model and the metamodel-based searching criterion, PGLO better explores the response surface, while with the application of pattern search, PGLO efficiently exploits the promising area for a better solution. Here we compare PGLO with Random Search (RS) and CGLO on a single processor.

The S^2TA system is run until a potential conflict is detected. At this point, we run PGLO, RS and CGLO to obtain the optimal alternate trajectory. The solution space is

discretized into 50×50 grids in both positive and negative polarity for θ_1 and θ_2 , and the speeds v_1 and v_2 in 15×15 grids, which gives an estimation of minimum probability of $p^* = 3.6165 \times 10^{-4}$ at $x^* = [-57.6000, -52.8000, 5, 11]$. In PGLO, we set a maximum of 10 alternative trajectories to be evaluated successively through each stream of the pattern search. Due to the high nonstationarity in the response of probability, each local region includes multiple local optimal solutions, and a maximum iteration of pattern search avoids the local search stage from overexploiting a sub-optimal local area before it explores the other promising local regions.

A maximum time for the S^2TA system to search for the alternative trajectory with all three algorithms is set at 5 minutes before termination, at which point, the final best solution is determined. Table 5.7 presents the numbers of evaluation points, the distances between the 'optimal' solution y_{true} and the observed best solution by the each approach $y_{approach}$.

Table 5.7: The number of objective function evaluations and the deviation of the optimal probability of conflict $y_{approach}$ found by each optimization algorithm to the true optimal

Approach	Number of Evaluation Points	$\ y_{approach} - y_{true}\ $
RS	2000	0.0019
CGLO	611	9.4876×10^{-4}
PGLO(1)	1421	4.5247×10^{-4}
PGLO(4)	3894	3.6214×10^{-4}

As the simulation runs fast, the RS was able to search at 2000 design points and identify an optimal solution that is close to the true optimal location. Although with one processor, none of the approaches managed to locate the optimal solution in the limited time, both CGLO and PGLO were able to locate a safer solution (with a 50% lower probability of conflict) than RS. As the iteration progresses, PGLO required approximately additional 5 minutes (10 minutes in total) to identify the optimal solution, while CGLO required additional 15 minutes (20 minutes in total) to get the optimal solution. This shows that CGLO works less efficiently than PGLO as the data size gets larger.

In this example where the simulation runs fast, a large part of the computational time in CGLO is spent on the computation of the model predictions and selecting next evalu-

5.7. A NAVIGATIONAL SAFETY PROBLEM

ation points. Hence only 611 points are evaluated to search the domain space and they do not sufficiently identify the global optimal area. The PGLO, incorporated with a fast pattern search, is more computationally efficient instead. It reduces the computational time of model predictions for more simulation evaluations and better explores the entire space with these additional simulation evaluations.

As PGLO has a parallel framework, running PGLO on the same personal computer can fully utilize its 4 processors. We see from Table 5.7 (last row) that PGLO with 4 processors is able to locate the optimal solution within 5 minutes of time constraint. Next, we compared the parallel realization of PGLO with RS and multi-start pattern search on four processors. For RS, we randomly selected four points for evaluation in each iteration. For multi-start pattern search, we initialized the pattern search from four Latin Hypercube designs and continued the search independently on four processors. A maximum budget of 1,000 replications was allowed in each iteration before we re-initialized the pattern search (avoid overexploiting a sub-optimal area). The results were averaged over 30 additional detected conflicts. With additional processors, much more alternative trajectories could be evaluated. Thus, all the algorithms were able to identify the global optimal area. It is assumed that the algorithm finds the global optimal solution if the Euclidean distance between the observed optimal trajectory and the 'true' optimal trajectory $\|x_{true} - x_{approach}\| \leq 5$. Table 5.8 shows that among the 30 conflict scenarios, PGLO found the most global optimal solutions.

Table 5.8: Number and percentage of scenarios for which each algorithm converges to the global optimal solution with 4 processors within 5 minutes

Approach	RS	MultPPS	PGLO
Percentage	12/30(40%)	19/30 (63%)	24/30 (80%)

Overall, the results show that PGLO can be better incorporated into the S^2TA system to provide a better alternative trajectory with a lower probability of conflict within the limited response time. It distributes the computing budget to the promising local regions more effectively and searches for new evaluation points more efficiently. This attribute is very helpful for such real-time control systems with complex simulation models that show high nonstationarity.

5.8 Conclusions

In this chapter, we proposed a parallel global and local optimization algorithm by utilizing the global and local structures of CGLO. It iteratively incorporates locally convergent direct search methods for fast exploitation and computational efficiency improvement. We then derived a multi-point global expected improvement function that better explores the entire space for multiple promising local regions. The performance of CGLO and PGLO was compared on a simple one-dimensional example. The parallelization efficiency of PGLO was further studied on five test functions. We also applied PGLO to solve a practical navigational problem which aims to find the best trajectory with the lowest probability of conflict. The results provide invaluable insights in improving the S^2TA system.

Chapter 6

CONCLUSIONS AND FUTURE RESEARCH

In this thesis, we investigated both the modeling and optimization problems in stochastic simulation context. In this chapter, we conclude the study by summarizing the main results of our research, and giving directions for possible future research.

6.1 Summary

In this thesis, we first investigated the application of Gaussian Process model in approximating the response surface of computer models where large data sets are observed in Chapter 3. We discussed the limitation of Gaussian Process model and proposed an additive global and local Gaussian Process (AGLGP) model that is computationally efficient and enables more flexibility in the modeling of the systems whose response can dramatically change over the design space. This additive structure comprises of a global model with a small set of inducing points to capture the global trend and several local models to capture the residual process from the global model. We then proposed an approach to determine the local regions and inducing points which have not been previously addressed but assumed given. We proposed two model estimation methods to obtain the predictive distribution and maximum likelihood parameter estimators; namely, one stage estimation and two stage estimation methods. We have further shown several nice properties of the additive model, such as identifiability. Our model was then nu-

merically compared with other approximation methods. The results indicate that the proposed AGLGP model is a promising method for applications in simulation systems with nonstationarity. With its global and local structure and computational efficiency, the AGLGP model is especially well suited for the simulation optimization.

In the following Chapter 4, we applied the AGLGP model to stochastic simulation optimization problems. With the AGLGP model, we adopted the efficient global optimization (EGO) framework and the expected improvement (EI) to allocate the computing budget for experiment designs. However, the EGO framework works less efficiently as the iteration progresses and the number of evaluations observed gets large. We have proposed a combined iterative global and local search framework, leveraging on the global and local structures of the additive global and local Gaussian Process model, to solve this problem. The proposed framework tends to identify a promising local region in the global search stage and then search finely in the promising region in the local search stage. We also proposed an allocation strategy that intelligently allocates budget to the evaluated points for the purpose of improving the metamodel fit and estimating the optimal solution. The property of global convergence was then analyzed. We tested the proposed framework with a simple one-dimensional example, followed by comparison with other metamodel-based optimization algorithms. The results from a more realistic navigational safety problem suggested that the proposed combined global and local optimization algorithm is a more reasonable method when optimizing a less expensive simulation model with highly nonstationary response surface than traditional methods.

In Chapter 5, we extended the combined global and local optimization method into a parallel environment. Although this method has been shown to perform efficiently in many situations, in some real-time control systems whose behaviour can dynamically change over time, a much faster optimization algorithm may be required to effectively respond to these system changes. We proposed a parallel global and local optimization approach that identifies multiple promising local regions in the global search stage and parallelizes the local search in each promising local region for simultaneous evaluations. The proposed multi-point global expected improvement function is able to better explore the entire space for multiple promising local regions before exploiting multiple local optimal solutions in a single local region. As the design points tend to be clus-

tered in promising local regions, the local search stage can become expensive in local regions with dense design points. We further incorporated the pattern search for faster exploitation to reduce the computational time of the local search stage. The global convergence was then studied. We illustrated the parallel approach with several examples and provided invaluable insights in improving the practical S^2TA system.

6.2 Future Research

The current work can be extended in the following directions.

1. In Chapter 3, even though the discontinuity of the AGLGP model can be solved by adding additional constraints between local models, the model estimation may become more expensive. Future work includes building a smooth structure for the AGLGP model through a dynamic or an overlapping local decomposition strategy.
2. The proposed AGLGP model focuses on improving the computational efficiency of the model estimation with large data sets. It may not be able to handle high input dimensions, especially when the number of local regions increases. Hence alternative approximation methods can be explored to reduce the input dimensions.
3. The optimization methods in Chapters 4 and 5 assume a constant allocation budget. Developing an adaptive scheme that dynamically determines the allocation budget for better evaluation in each iteration is worth further study for stochastic simulation systems. For example, increasing the allocation budget over iterations may help to improve the convergence rate.
4. In the proposed metamodeling method and optimization algorithms, Gaussian Process model is used as the metamodel for its flexibility and its unique statistical view of prediction error. However, comparison with other metamodels and metamodel-based optimization algorithms are also important issues to be addressed in future work.
5. The parallel framework in Chapter 5 aims at defining a fast searching criterion for parallel simulations. A more comprehensive comparison with other parallel

optimization methods is worth further study. The framework can also be further accelerated by parallelizing the model estimation and the searching criterion.

Bibliography

- Ankenman, B., Nelson, B. L., and Staum, J. (2010). Stochastic kriging for simulation metamodeling. *Operations Research*, 58(2):371–382. 4, 24, 26, 27
- Arora, J. S. (2004). *Introduction to optimum design*. Academic Press. 10
- Ba, S. and Joseph, V. R. (2012). Composite Gaussian process models for emulating expensive functions. *The Annals of Applied Statistics*, 6(4):1838–1860. 5, 37
- Baker, C. T. (1977). *The numerical treatment of integral equations*. Clarendon Press. 17
- Banerjee, S., Gelfand, A. E., Finley, A. O., and Sang, H. (2008). Gaussian predictive process models for large spatial data sets. *Journal of the Royal Statistical Society. Series B, Statistical methodology*, 70(4):825–848. 17, 44, 46
- Barton, R. R. and Meckesheimer, M. (2006). Metamodel-based simulation optimization. *Handbooks in operations research and management science*, 13:535–574. 6, 7
- Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. springer. 3
- Bonabeau, E. (2002). Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 99(suppl 3):7280–7287. 2
- Booker, A. J., Dennis Jr, J., Frank, P. D., Serafini, D. B., Torczon, V., and Trosset, M. W. (1999). A rigorous framework for optimization of expensive functions by surrogates. *Structural optimization*, 17(1):1–13. 79
- Box, G. E., Draper, N. R., et al. (1987). *Empirical model-building and response surfaces*, volume 424. Wiley New York. 20
- Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). *Classification and Regression Trees*. Wadsworth. 56
- Buhmann, M. D. (2003). *Radial Basis Functions: Theory and Implementations*, volume 12. Cambridge University Press. 14
- Celis, M., Dennis, J., and Tapia, R. (1985). A trust region strategy for nonlinear equality constrained optimization. *Numerical optimization*, 1984:71–82. 20
- Chen, C.-H. and Lee, L. H. (2010). *Stochastic Simulation Optimization: An Optimal Computing Budget Allocation*. World Scientific. 8, 62, 65, 92, 93
- Chevalier, C., Bect, J., Ginsbourger, D., Vazquez, E., Picheny, V., and Richet, Y. (2014). Fast parallel kriging-based stepwise uncertainty reduction with application to the identification of an excursion set. *Technometrics*, 56(4):455–465. 78

- Chevalier, C. and Ginsbourger, D. (2013). Fast computation of the multi-points expected improvement with applications in batch selection. In *International Conference on Learning and Intelligent Optimization*, pages 59–69. Springer. 78
- Clark, S. C. and Frazier, P. I. (2012). Parallel global optimization using an improved multi-points expected improvement criterion. In *INFORMS Optimization Society Conference, Miami*. 78
- Clarke, S. M., Griebisch, J. H., and Simpson, T. W. (2005). Analysis of support vector regression for approximation of complex engineering analyses. *Journal of mechanical design*, 127(6):1077–1087. 15
- Conn, A. R., Scheinberg, K., and Toint, P. L. (1997). Recent progress in unconstrained nonlinear optimization without derivatives. *Mathematical programming*, 79(1-3):397–414. 21
- Contal, E., Buffoni, D., Robicquet, A., and Vayatis, N. (2013). Parallel Gaussian process optimization with upper confidence bound and pure exploration. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 225–240. Springer. 78
- Cressie, N. (1993). *Statistics for Spatial Data*. Wiley, New York. 3, 4, 15, 24
- Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D. (1991). 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. 1
- Davidsson, P., Henesey, L., Ramstedt, L., Törnquist, J., and Wernstedt, F. (2005). An analysis of agent-based approaches to transport logistics. *Transportation Research part C: emerging technologies*, 13(4):255–271. 2
- Desautels, T., Krause, A., and Burdick, J. (2012). Parallelizing exploration-exploitation tradeoffs with Gaussian process bandit optimization. *Journal of Machine Learning Research*, 15:3873–3923. 78
- Duffuaa, S. and Andijani, A. (1999). An integrated simulation model for effective planning of maintenance operations for Saudi Arabian Airlines (SAUDIA). *Production Planning & Control*, 10(6):579–584. 1
- Fonseca, D., Navarrese, D., and Moynihan, G. (2003). Simulation metamodeling through artificial neural networks. *Engineering Applications of Artificial Intelligence*, 16(3):177–183. 16
- Fonseca, D. J. and Navarrese, D. (2002). Artificial neural networks for job shop simulation. *Advanced Engineering Informatics*, 16(4):241–246. 16
- Friedman, J. H. (1991). Multivariate adaptive regression splines. *The annals of statistics*, pages 1–67. 14
- Furrer, R., Genton, M. G., and Nychka, D. (2006). Covariance tapering for interpolation of large spatial datasets. *Journal of Computational and Graphical Statistics*, 15(3):502–523. 18
- Gano, S. E., Renaud, J. E., Martin, J. D., and Simpson, T. W. (2006). Update strategies for kriging models used in variable fidelity optimization. *Structural and Multidisciplinary Optimization*, 32(4):287–298. 21

BIBLIOGRAPHY

- Ginsbourger, D., Le Riche, R., and Carraro, L. (2008). A multi-points criterion for deterministic parallel global optimization based on Gaussian processes. 78, 80
- Ginsbourger, D., Le Riche, R., and Carraro, L. (2010). Kriging is well-suited to parallelize optimization. In *Computational Intelligence in Expensive Optimization Problems*, pages 131–162. Springer. 85
- Glasserman, P. (2003). *Monte Carlo methods in financial engineering*, volume 53. Springer Science & Business Media. 1
- Gramacy, R. B. and Apley, D. W. (2014). Local Gaussian process approximation for large computer experiments. *Journal of Computational and Graphical Statistics*, pages 1–28. 18
- Gramacy, R. B. and Digabel, S. L. (2015). The mesh adaptive direct search algorithm with treed Gaussian process surrogates. *Pacific Journal of Optimization*, 11(03):419–447. 79
- Gramacy, R. B. and Haaland, B. (2016). Speeding up neighborhood search in local Gaussian process prediction. *Technometrics*, 58(3):294–303. 18
- Gramacy, R. B. and Lee, H. K. H. (2008). Bayesian treed Gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483):1119–1130. 5
- Gramacy, R. B. and Lee, H. K. H. (2009). Adaptive design and analysis of supercomputer experiments. *Technometrics*, 51(2):130–145. 78
- Gray, G. A., Martinez-Canales, M., Lee, H. K., Taddy, M., and Gramacy, R. B. (2007). Enhancing parallel pattern search optimization with a Gaussian process oracle. *Proceedings of the 14th Nuclear Explosive Codes Development Conference (NECDC)*. 79
- Hardy, R. L. (1971). Multiquadric equations of topography and other irregular surfaces. *Journal of geophysical research*, 76(8):1905–1915. 13
- Hong, L. J. and Nelson, B. L. (2006). Discrete optimization via simulation using COMPASS. *Operations Research*, 54(1):115–129. 6, 51
- Hu, W., Yao, L. G., and Hua, Z. Z. (2008). Optimization of sheet metal forming processes by adaptive response surface based on intelligent sampling method. *Journal of materials processing technology*, 197(1):77–88. 79
- Huang, D., Allen, T. T., Notz, W. I., and Zeng, N. (2006). Global optimization of stochastic black-box systems via sequential kriging meta-models. *Journal of Global Optimization*, 34(3):441–466. 7, 51
- Hussain, M. F., Barton, R. R., and Joshi, S. B. (2002). Metamodeling: radial basis functions, versus polynomials. *European Journal of Operational Research*, 138(1):142–154. 14
- Jin, R., Chen, W., and Simpson, T. W. (2001). Comparative studies of metamodelling techniques under multiple modelling criteria. *Structural and Multidisciplinary Optimization*, 23(1):1–13. 15

- Jones, D. (2001). A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21(4):345–383. 51, 53
- Jones, D., Schonlau, M., and Welch, W. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492. 7, 16, 21, 24, 51, 52, 53, 78, 84
- Kansa, E. J. (1990). Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics—II solutions to parabolic, hyperbolic and elliptic partial differential equations. *Computers & mathematics with applications*, 19(8):147–161. 14
- Kilmer, R. A., Smith, A. E., and Shuman, L. J. (1997). An emergency department simulation and a neural network metamodel. *Journal of the Society for Health Systems*, 5(3):63–79. 16
- Kim, H.-M., Mallick, B. K., and Holmes, C. C. (2005). Analyzing nonstationary spatial data using piecewise Gaussian processes. *Journal of the American Statistical Association*, 100(470):653–668. 5
- Kleijnen, J. P. (1998). Experimental design for sensitivity analysis, optimization, and validation of simulation models. *Handbook of simulation*, pages 173–223. 13
- Kleijnen, J. P. (2008). *Design and analysis of simulation experiments*, volume 20. Springer. 2
- Kleijnen, J. P. (2014). Simulation-optimization via kriging and bootstrapping: a survey. *Journal of Simulation*, 8(4):241–250. 24
- Kleijnen, J. P., Den Hertog, D., and Angün, E. (2004). Response surface methodology’s steepest ascent and step size revisited. *European Journal of Operational Research*, 159(1):121–131. 7
- Kleijnen, J. P., van Beers, W., and Van Nieuwenhuysse, I. (2012). Expected improvement in efficient global optimization through bootstrapped kriging. *Journal of global optimization*, 54(1):59–73. 22
- Koopmans, T. C. and Reiersol, O. (1950). The identification of structural characteristics. *The Annals of Mathematical Statistics*, pages 165–181. 42
- Larsson, E. and Fornberg, B. (2003). A numerical study of some radial basis function based solution methods for elliptic pdes. *Computers & Mathematics with Applications*, 46(5):891–902. 14
- Li, Y., Ng, S., Xie, M., and Goh, T. (2010). A systematic comparison of metamodeling techniques for simulation optimization in Decision Support Systems. *Applied Soft Computing*, 10(4):1257–1273. 3, 23
- MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. volume 1. 44
- Montgomery, D. C. (1991). *Design and analysis of experiments*. John Wiley & Sons. 79

BIBLIOGRAPHY

- Nasreddin, M. and Mollaghasemi, M. (1999). The development of a methodology for the use of neural networks and simulation modeling in system design. In *Simulation Conference Proceedings, 1999 Winter*, volume 1, pages 537–542. IEEE. 16
- Ng, S. H. and Yin, J. (2012). Bayesian kriging analysis and design for stochastic simulations. *ACM Transactions on Modeling and Computer Simulation*, 22(3):1–26. 26, 35
- Okabe, A., Boots, B., Sugihara, K., and Chiu, S. N. (2009). *Spatial tessellations: concepts and applications of Voronoi diagrams*, volume 501. John Wiley & Sons. 44
- Park, C., Huang, J., and Ding, Y. (2011). Domain decomposition approach for fast Gaussian process regression of large spatial data sets. *The Journal of Machine Learning Research*, 12:1697–1728. 18, 38
- Pedrielli, G., Xing, Y., Peh, J. H., and Ng, S. H. A real time simulation optimization framework for vessel collision avoidance and the case of singapore strait. *submitted to IEEE Transactions on Intelligent Transportation Systems*. 10, 71, 78
- Picheny, V., Ginsbourger, D., Richet, Y., and Caplin, G. (2013). Quantile-based optimization of noisy computer experiments with tunable precision. *Technometrics*, 55(1):2–13. 7, 51, 52, 70
- Powell, M. J. (2002). UOBYQA: unconstrained optimization by quadratic approximation. *Mathematical Programming*, 92(3):555–582. 21
- Powell, M. J. (2003). On trust region methods for unconstrained minimization without derivatives. *Mathematical programming*, 97(3):605–623. 21
- Quan, N., Yin, J., Ng, S. H., and Lee, L. H. (2013). Simulation optimization via kriging: a sequential search using expected improvement with computing budget constraints. *IIE Transactions*, 45(7):763–780. 7, 51, 52, 53, 58, 59, 70, 90, 103
- Quiñonero Candela, J. and Rasmussen, C. (2005). A unifying view of sparse approximate Gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959. 17
- Rao, S. S. (1996). *Engineering optimization*. Wiley, New York. 10
- Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian processes for machine learning*. MIT Press, Boston. 3, 4, 17
- Regis, R. G. and Shoemaker, C. a. (2007). A stochastic radial basis function method for the global optimization of expensive functions. *INFORMS Journal on Computing*, 19(4):497–509. 22, 56, 64, 91, 94
- Ruppert, D. (2011). *Statistics and data analysis for financial engineering*. Springer. 13
- Sacks, J., Welch, W., Mitchell, T., and Wynn, H. (1989). Design and analysis of computer experiments. *Statistical science*, 4(4):409–435. 3, 4, 15, 16, 24, 84
- Sang, H. and Huang, J. (2012). A full scale approximation of covariance functions for large spatial data sets. *Journal of the Royal Statistical Society*, 74:111–132. 4, 18, 24, 44, 46

- Santner, T. J., Brian J. Williams, and Notz, W. I. (2003). *The Design and Analysis of Computer Experiments*. Springer Science & Business Media. 2, 3, 4, 15, 24, 84
- Sasena, M. J., Papalambros, P., and Goovaerts, P. (2002). Exploration of metamodeling sampling criteria for constrained global optimization. *Engineering optimization*, 34(3):263–278. 22
- Schonlau, M., Welch, W. J., and Jones, D. R. (1998). Global versus local search in constrained optimization of computer models. *Lecture Notes-Monograph Series*, pages 11–25. 84
- Shi, L. and Ólafsson, S. (2000). Nested partitions method for global optimization. *Operations Research*, 48(3):390–407. 6, 51
- Simpson, T. W., Poplinski, J., Koch, P. N., and Allen, J. K. (2001). Metamodels for computer-based engineering design: survey and recommendations. *Engineering with computers*, 17(2):129–150. 3, 23
- Snelson, E. and Ghahramani, Z. (2005). Sparse Gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pages 1257–1264. 39
- Snelson, E. and Ghahramani, Z. (2007). Local and global sparse Gaussian process approximations. *International Conference on Artificial Intelligence and Statistics*. 4, 19, 24, 44, 46
- Sóbestor, A., Leary, S. J., and Keane, A. J. (2004). A parallel updating scheme for approximating and optimizing high fidelity computer simulations. *Structural and Multidisciplinary Optimization*, 27(5):371–383. 78
- Sóbestor, A., Leary, S. J., and Keane, A. J. (2005). On the design of optimization strategies based on global response surface approximation models. *Journal of Global Optimization*, 33(1):31–59. 22
- Sun, C., Jin, Y., Zeng, J., and Yu, Y. (2015). A two-layer surrogate-assisted particle swarm optimization algorithm. *Soft Computing*, 19(6):1461–1475. 79
- Sun, L., Hong, L. J., and Hu, Z. (2014). Balancing exploitation and exploration in discrete optimization via simulation through a Gaussian process-based search. *Operations Research*, 62(6):1416–1438. 69
- Surjanovic, S. and Bingham, D. Virtual library of simulation experiments: Test functions and datasets. Retrieved December 31, 2016, from <http://www.sfu.ca/~ssurjano>. 100
- Taddy, M. A., Lee, H. K. H., Gray, G. A., and Griffin, J. D. (2009). Bayesian guided pattern search for robust local optimization. *Technometrics*, 51(4):389–401. 79
- Torczon, V. (1997). On the convergence of pattern search algorithms. *SIAM Journal on optimization*, 7(1):1–25. 6, 81, 85, 92
- Torn, A. and Zilinskas, A. (1989). *Global optimization*. Springer-Verlag New York, Inc. 64, 94
- Wang, G. G., Dong, Z., and Aitchison, P. (2001). Adaptive response surface method—a global optimization scheme for approximation-based design problems. *Engineering Optimization*, 33(6):707–733. 79

BIBLIOGRAPHY

- Wild, S. M., Regis, R. G., and Shoemaker, C. A. (2008). ORBIT: Optimization by radial basis function interpolation in trust-regions. *SIAM Journal on Scientific Computing*, 30(6):3197–3219. 21
- Xiong, Y., Chen, W., Apley, D., and Ding, X. (2007). A non-stationary covariance-based kriging method for metamodeling in engineering design. *International Journal for Numerical Methods in Engineering*, 71(6):733–756. 27
- Xu, J., Nelson, B. L., and Hong, L. J. (2010). Industrial strength COMPASS. *ACM Transactions on Modeling and Computer Simulation*, 20(1):1–29. 6
- Yin, J., Ng, S., and Ng, K. (2011). Kriging metamodel with modified nugget-effect: The heteroscedastic variance case. *Computers & Industrial Engineering*, 61(3):760–777. 4, 5, 24, 27, 44, 46, 53

Appendix A

Derivation of Predictive

Distribution in Section 3.3.1.1

Integrating conditional distribution of y_0 in equation (3.10) with (3.11) and (3.12)

$$\begin{aligned}
p(y|x_0, \mathbf{x}_g, \mathbf{x}, \mathbf{y}) &= \iint p(y|x_0, \mathbf{x}_g, \mathbf{y}_g, \mathbf{y}_1, \mathbf{x}, \mathbf{y}) p(\mathbf{y}_1, \mathbf{y}_g | \mathbf{x}_g, \mathbf{x}, \mathbf{y}) d\mathbf{y}_g d\mathbf{y}_1 \\
&= \iint p(y|x_0, \mathbf{x}_g, \mathbf{y}_g, \mathbf{y}_1, \mathbf{x}, \mathbf{y}) p(\mathbf{y}_1 | \mathbf{x}_g, \mathbf{y}_g, \mathbf{x}, \mathbf{y}) p(\mathbf{y}_g | \mathbf{x}_g, \mathbf{x}, \mathbf{y}) d\mathbf{y}_g d\mathbf{y}_1 \\
&\propto \iint \exp\left(-\frac{[y - (\mu + \mathbf{g}'\mathbf{G}_m^{-1}(\mathbf{y}_g - \mu) + \mathbf{l}'(\mathbf{L}_n + \Sigma_\epsilon)^{-1}\mathbf{y}_1)]^2}{2(\lambda + \gamma + \sigma_\epsilon^2)}\right) \\
&\quad \times \exp\left(-\frac{[\mathbf{y}_1 - \boldsymbol{\mu}_l]'\Sigma^{-1}[\mathbf{y}_1 - \boldsymbol{\mu}_l]}{2}\right) p(\mathbf{y}_g | \mathbf{x}_g, \mathbf{x}, \mathbf{y}) d\mathbf{y}_g d\mathbf{y}_1 \\
&\propto \int \exp\left(-\frac{[y - (\mu + \mathbf{g}'\mathbf{G}_m^{-1}(\mathbf{y}_g - \mu) + \mathbf{l}'[\Sigma_\epsilon + \mathbf{L}_n]^{-1}\boldsymbol{\mu}_l)]^2}{2(\lambda + \gamma + \mathbf{l}'[\mathbf{L}_n + \Sigma_\epsilon]^{-1}\Sigma[\mathbf{L}_n + \Sigma_\epsilon]^{-1}\mathbf{l} + \sigma_\epsilon^2)}\right) \\
&\quad \times \exp\left(-\frac{[\mathbf{y}_g - \boldsymbol{\mu}_g]'\mathbf{G}_m\mathbf{Q}_m^{-1}\mathbf{G}_m^{-1}[\mathbf{y}_g - \boldsymbol{\mu}_g]}{2}\right) d\mathbf{y}_g \\
&\propto \exp\left(-\frac{[y - \hat{y}(x_0)]^2}{2\hat{s}^2(x_0)}\right)
\end{aligned}$$

where $\boldsymbol{\mu}_g = \mu + \mathbf{G}_m\mathbf{Q}_m^{-1}\mathbf{G}_{mn}\mathbf{K}^{-1}(\mathbf{y} - \mu)$, $\boldsymbol{\mu}_l = \mathbf{L}_n\mathbf{K}^{-1}\{\mathbf{y} - \mu - \mathbf{G}_{nm}\mathbf{G}_m^{-1}(\mathbf{y}_g - \mu)\}$

and $\Sigma = \mathbf{L}_n - \mathbf{L}_n\mathbf{K}^{-1}\mathbf{L}_n + \Sigma_\epsilon$. So we have $p(y|x_0, \mathbf{x}_g, \mathbf{x}, \mathbf{y}) = N(\hat{y}(x_0), \hat{s}^2(x_0))$,

where

$$\hat{y}(x_0) = \mu + [\mathbf{g}'\mathbf{Q}_m^{-1}\mathbf{G}_{mn} + \mathbf{l}'(\mathbf{L}_n + \Sigma_\epsilon)^{-1}\mathbf{L}_n\mathbf{K}^{-1}(\mathbf{K} - \mathbf{G}_{nm}\mathbf{Q}_m^{-1}\mathbf{G}_{mn})]\mathbf{K}^{-1}(\mathbf{y} - \mu)$$

$$\hat{s}^2(x_0) = G_{nn} - \mathbf{g}'(\mathbf{G}_m^{-1} - \mathbf{Q}_m^{-1})\mathbf{g} + \frac{(1 - \mathbf{l}'\mathbf{G}_m^{-1}\mathbf{l})^2}{\mathbf{l}'\mathbf{G}_m^{-1}\mathbf{l}} + L_{nn} - \mathbf{l}'[\Sigma_\epsilon + \mathbf{L}_n]^{-1}\mathbf{L}_n\mathbf{K}^{-1}\mathbf{L}_n[\Sigma_\epsilon + \mathbf{L}_n]^{-1}\mathbf{l}$$

Appendix B

Proof of Theorem 3.1

For the AGLGP model, the predictive distribution derived in (3.10) gives a mean of

$$\hat{y}(x_0) = \mu + [\mathbf{g}'\mathbf{Q}_m^{-1}\mathbf{G}_{mn} + \mathbf{l}'(\mathbf{L}_n + \Sigma_\epsilon)^{-1}\mathbf{L}_n\mathbf{K}^{-1}(\mathbf{K} - \mathbf{G}_{nm}\mathbf{Q}_m^{-1}\mathbf{G}_{mn})]\mathbf{K}^{-1}(\mathbf{y} - \mu) \quad (\text{B.1})$$

where $\mathbf{Q}_m = \mathbf{G}_m + \mathbf{G}_{mn}\mathbf{K}^{-1}\mathbf{G}_{nm}$ and $\mathbf{K} = \mathbf{L}_n + \Lambda + \Sigma_\epsilon$. The predictive mean can also be expressed by

$$\hat{y}(x_0) = \mu + [\mathbf{g}'\mathbf{G}_m^{-1}\mathbf{G}_{mn} + \mathbf{l}'(\mathbf{L}_n + \Sigma_\epsilon)^{-1}\mathbf{L}_n]\mathbf{R}^{-1}(\mathbf{y} - \mu) \quad (\text{B.2})$$

where $\mathbf{R} = \mathbf{G}'_{mn}\mathbf{G}_m^{-1}\mathbf{G}_{mn} + \Lambda + \mathbf{L}_n + \Sigma$ and $\hat{\mu} = \frac{\mathbf{1}'\mathbf{R}^{-1}\mathbf{y}}{\mathbf{1}'\mathbf{R}^{-1}\mathbf{1}}$. First we show equation (B.1) and equation (B.2) are equivalent. From the Woodbury identity, equation (B.2) becomes

$$\begin{aligned} \hat{y}(x_0) &= \mu + [\mathbf{g}'\mathbf{G}_m^{-1}\mathbf{G}_{mn} + \mathbf{l}'(\mathbf{L}_n + \Sigma_\epsilon)^{-1}\mathbf{L}_n](\mathbf{I} - \mathbf{K}^{-1}\mathbf{G}_{nm}\mathbf{Q}_m^{-1}\mathbf{G}_{mn})\mathbf{K}^{-1}(\mathbf{y} - \mu) \\ &= \mu + \{\mathbf{g}'\mathbf{G}_m^{-1}\mathbf{G}_{mn}(\mathbf{I} - \mathbf{K}^{-1}\mathbf{G}_{nm}\mathbf{Q}_m^{-1}\mathbf{G}_{mn}) \\ &\quad + \mathbf{l}'(\mathbf{L}_n + \Sigma_\epsilon)^{-1}\mathbf{L}_n(\mathbf{I} - \mathbf{K}^{-1}\mathbf{G}_{nm}\mathbf{Q}_m^{-1}\mathbf{G}_{mn})\}\mathbf{K}^{-1}(\mathbf{y} - \mu) \\ &= \mu + \{\mathbf{g}'\mathbf{G}_m^{-1}\mathbf{G}_{mn} - \mathbf{g}'\mathbf{G}_m^{-1}\mathbf{G}_{mn}\mathbf{K}^{-1}\mathbf{G}_{nm}\mathbf{Q}_m^{-1}\mathbf{G}_{mn} \\ &\quad + \mathbf{l}'(\mathbf{L}_n + \Sigma_\epsilon)^{-1}\mathbf{L}_n\mathbf{K}^{-1}(\mathbf{K} - \mathbf{G}_{nm}\mathbf{Q}_m^{-1}\mathbf{G}_{mn})\}\mathbf{K}^{-1}(\mathbf{y} - \mu) \\ &= \mu + \{\mathbf{g}'(\mathbf{G}_m^{-1} - \mathbf{G}_m^{-1}(\mathbf{I} - \mathbf{G}_m\mathbf{Q}_m^{-1}))\mathbf{G}_{mn} \\ &\quad + \mathbf{l}'(\mathbf{L}_n + \Sigma_\epsilon)^{-1}\mathbf{L}_n\mathbf{K}^{-1}(\mathbf{K} - \mathbf{G}_{nm}\mathbf{Q}_m^{-1}\mathbf{G}_{mn})\}\mathbf{K}^{-1}(\mathbf{y} - \mu) \end{aligned}$$

$$= \mu + \{ \mathbf{g}' \mathbf{Q}_m^{-1} \mathbf{G}_{mn} + \mathbf{l}' (\mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon)^{-1} \mathbf{L}_n \mathbf{K}^{-1} (\mathbf{K} - \mathbf{G}_{nm} \mathbf{Q}_m^{-1} \mathbf{G}_{mn}) \} \mathbf{K}^{-1} (\mathbf{y} - \boldsymbol{\mu})$$

which shows that equation (B.1) and equation (B.2) are equivalent. Based on equation (B.2), $\hat{y}(x_0)$ is also a linear combination of \mathbf{y} , *i.e.* $\hat{y}(x_0) = \sum_{i=1}^n \lambda_i y_i$,

$$\lambda_i = \left[\frac{\mathbf{1}' \mathbf{R}^{-1}}{\mathbf{1}' \mathbf{R}^{-1} \mathbf{1}} + (\mathbf{g}' \mathbf{G}_m^{-1} \mathbf{G}_{mn} + \mathbf{l}' [\mathbf{L}_n + \boldsymbol{\Sigma}_\epsilon]^{-1} \mathbf{L}_n) \mathbf{R}^{-1} (1 - \mathbf{1}' \frac{\mathbf{1}' \mathbf{R}^{-1}}{\mathbf{1}' \mathbf{R}^{-1} \mathbf{1}}) \right] e_i$$

where $e_i = [0, 0, \dots, \underbrace{1}_{\text{the } i\text{th element}}, \dots, 0, 0]$; $\sum_{i=1}^n \lambda_i = 1$. $E[\hat{y}(x_0)] = E[\sum_{i=1}^n \lambda_i y_i] = \sum_{i=1}^n \lambda_i E[y_i] = E[y(x_0)]$.

Appendix C

Proof of Proposition 3.2

In this case the conditional likelihood of observations \mathbf{y} is given by

$$\mathbf{y}|x, \mathbf{y}_g, \mathbf{x} \sim N(\boldsymbol{\mu} + \mathbf{G}'_n \mathbf{G}_n^{-1}(\mathbf{y}_g - \mathbf{1}'\boldsymbol{\mu}), \boldsymbol{\Sigma}_\epsilon),$$

The conditional distribution of \mathbf{y}_g given \mathbf{y} is

$$\mathbf{y}_g|\mathbf{x}, \mathbf{y} \sim N((\mathbf{G}_n^{-1} + \boldsymbol{\Sigma}_\epsilon^{-1})^{-1}\{\boldsymbol{\Sigma}_\epsilon^{-1}\mathbf{y} + \mathbf{G}_n^{-1}\mathbf{1}'\boldsymbol{\mu}\}, (\mathbf{G}_n^{-1} + \boldsymbol{\Sigma}_\epsilon^{-1})^{-1}),$$

so the predictive distribution of y_0 at any point x_0 can be derived as

$$y_0|x_0, \mathbf{x}, \mathbf{y} \sim N(\hat{y}(x_0), \hat{s}^2(x_0)) \tag{C.1}$$

where $\hat{y}(x_0) = \boldsymbol{\mu} + \mathbf{g}'(\mathbf{G}_n + \boldsymbol{\Sigma}_\epsilon)^{-1}(\mathbf{y} - \mathbf{1}'\boldsymbol{\mu})$ and $\hat{s}^2(x_0) = G_{nn} - \mathbf{g}'(\mathbf{G}_n + \boldsymbol{\Sigma}_\epsilon)^{-1}\mathbf{g}$.

This is equivalent to the Stochastic Kriging and MNEK predictor and variance in (3.2) and (3.3).

Appendix D

Proof of Identifiability in Section

3.4

Lemma 3.8. For $\forall x_i \in \mathbf{D}_p$ and $\forall x_j \in \mathbf{D}_q$ and $p \neq q$, the $(ij)_{th}$ element in covariance matrix $R_{ij}(\phi^1) = R_{ij}(\phi^2)$ only satisfied when $\sigma_1^2 = \sigma_2^2$ and $\theta_1 = \theta_2$

Proof. Given the expression of R_{ij} , it suffices to show that $\forall x_i, x_j$ in different local regions, $G_{ij}(\sigma_1^2, \theta_1) = G_{ij}(\sigma_2^2, \theta_2)$ only when $\theta_1 = \theta_2$ and $\sigma_1^2 = \sigma_2^2$. We can further simplify $G_{ij} = \sigma^2 h(|x_i - x_j|, \theta)$, where h is a function of θ and Euclidean distance between x_i and x_j . If there exist two set of parameters $\theta_1 \neq \theta_2$ and $\sigma_1^2 \neq \sigma_2^2$ that satisfies $G_{ij}(\sigma_1^2, \theta_1) = G_{ij}(\sigma_2^2, \theta_2)$ given $|x_i - x_j|$, i.e. $\sigma_1^2 h(|x_i - x_j|, \theta_1) = \sigma_2^2 h(|x_i - x_j|, \theta_2)$, we have

$$\frac{\sigma_1^2}{\sigma_2^2} = \frac{h(|x_i - x_j|, \theta_2)}{h(|x_i - x_j|, \theta_1)} \quad (\text{D.1})$$

Then for a different pair of evaluation point x_i, x_k that $|x_i - x_j| \neq |x_i - x_k|$, since h is not a linear function of θ , so we have

$$\frac{h(|x_i - x_k|, \theta_2)}{h(|x_i - x_k|, \theta_1)} \neq \frac{h(|x_i - x_j|, \theta_2)}{h(|x_i - x_j|, \theta_1)} \quad (\text{D.2})$$

So $G_{ik}(\sigma_1^2, \theta_1) = G_{ik}(\sigma_2^2, \theta_2)$ is not satisfied.

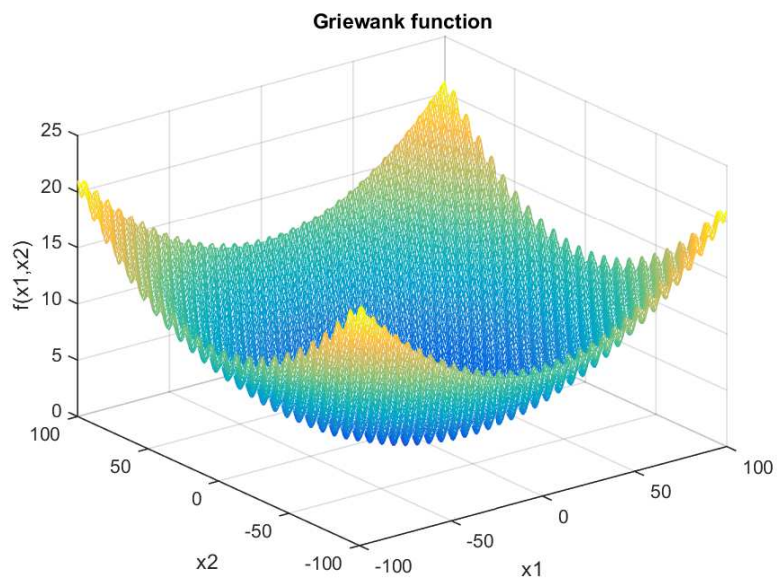
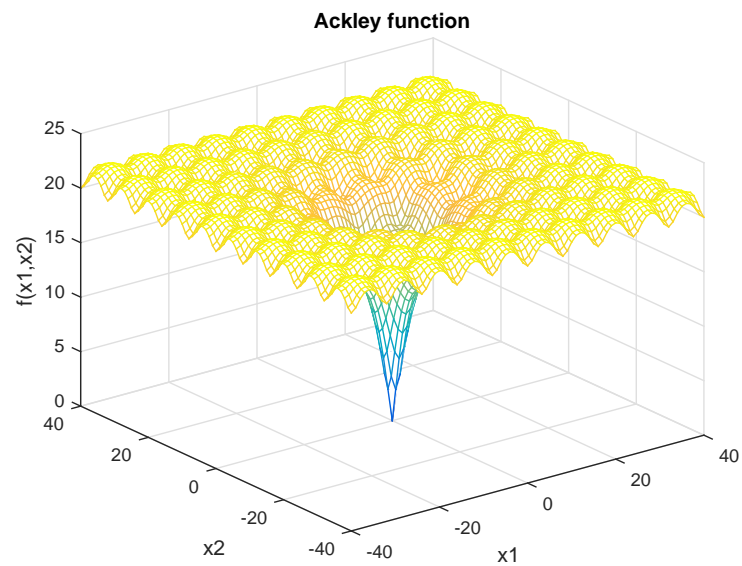
□

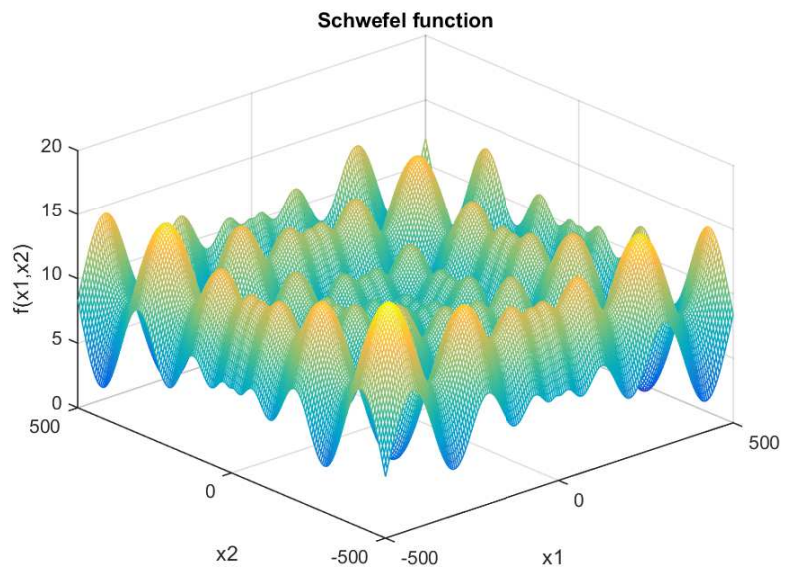
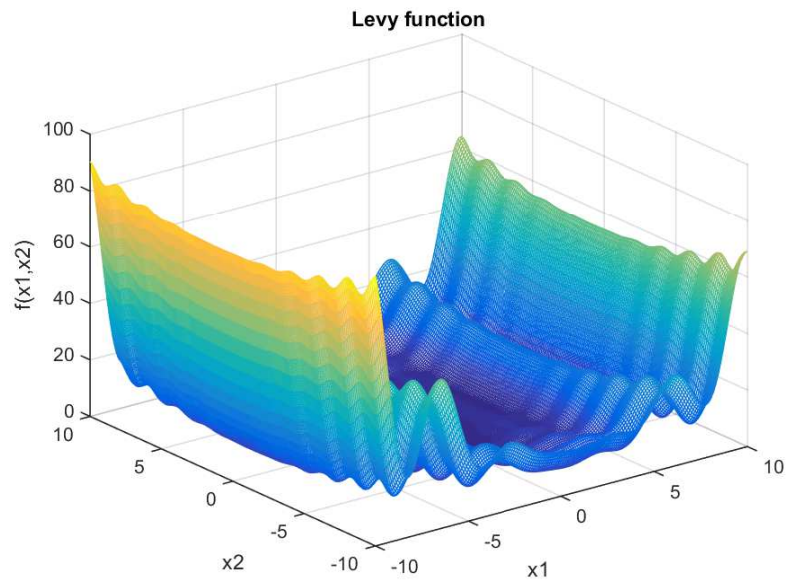
Lemma 3.9 Given $\sigma_1^2 = \sigma_2^2$ and $\theta_1 = \theta_2$, for $\forall x_i, x_j \in \mathbf{D}_k$, $R_{ij}(\phi^1) = R_{ij}(\phi^2)$ only satisfied when $\tau_1^2 = \tau_2^2$ and $\alpha_1 = \alpha_2$

Proof. Given $\sigma_1^2 = \sigma_2^2$ and $\boldsymbol{\theta}_1 = \boldsymbol{\theta}_2$, we have $G_{ij}(\sigma_1^2, \boldsymbol{\theta}_1) = G_{ij}(\sigma_2^2, \boldsymbol{\theta}_2)$ everywhere. When $x_i \in \mathbf{D}_p, x_j \in \mathbf{D}_q, p = q$, $R_{ij} = G_{ij} + L_{ij}$, so we have $R_{ij}(\phi^1) = R_{ij}(\phi^2)$ only when $L_{ij}(\tau_1^2, \boldsymbol{\alpha}_1) = L_{ij}(\tau_2^2, \boldsymbol{\alpha}_2)$. Now we can say that $\boldsymbol{\mu}(\phi^1) = \boldsymbol{\mu}(\phi^2)$ and $\mathbf{R}(\phi^1) = \mathbf{R}(\phi^2)$ is only satisfied when $\phi^1 = \phi^2$. So the AGLGP model is identifiable. \square

Appendix E

Test Functions in Section 5.6.2





LIST OF PUBLICATIONS

Meng, Q., & Ng, S. H. Additive Global and Local Gaussian Process Model for Simulation Metamodeling and Optimization, submitted.

Meng, Q., & Ng, S. H. Parallel global and local method for stochastic simulation optimization with an AGLGP model, working paper.

Meng, Q., & Ng, S. H. (2016) Combined global and local method for stochastic simulation optimization with an AGLGP model, *Proceedings of the 2016 Winter Simulation Conference*, IEEE Press.

Meng, Q., & Ng, S. H. (2015) An additive global and local gaussian process model for large data sets, *Proceedings of the 2015 Winter Simulation Conference*, IEEE Press, 505-516.