UNIVERSITY OF TEXAS AT ARLINGTON

DEPARTMENT OF INDUSTRIAL, MANUFACTURING, AND SYSTEMS ENGINEERING

PH.D. DISSERTATION

TK-MARS: An Efficient Approach for Deterministic and Stochastic Black-Box Optimization

By:
HADIS ANAHIDEH

Advisors:
DR. JAY ROSENBERGER AND DR. VICTORIA CHEN

Committee Members:
DR. MANFRED HUBER
DR. YUAN ZHOU

AUGUST-2018

In memory of my father Pariz Anahideh

& To my mother Elaheh Babaie

With love and eternal appreciation

Acknowledgements

I would like to express the deepest appreciation to my supervising professors Dr. Jay Rosenberger and Dr. Victoria Chen for their constant support, patience, and invaluable advice. They are the most brilliant people I have ever worked with. I want to extend my appreciation to my kind committee members, Dr. Manfred Huber and Dr. Yuan Zhou. I also want to thank Dr. Bill Corley, Dr. Sheik Imrhan, and the IMSE department for the support during my Ph.D.

I cannot be grateful enough to my parents who sacrificed their lives for my success. Thank you to my brother Mohammadhosain for being way more than my brother, and my best friend and husband Abolfazl for his persistent help during my Ph.D.

Abstract

Surrogate optimization approaches for black-box functions focus on approximating the underlying function, using metamodeling techniques, in order to optimize computationally expensive simulation models.

Historically, surrogate optimization models have been validated by deterministic (noiseless) functions with every variable being significant. As a result, many surrogate optimization models used interpolating surrogates. However, many real-world experiments often times include parameters that are insignificant and uncertainties associated with the black-box function. Using traditional interpolating surrogate optimization methods can lead to surrogate models with unnecessary predictors and sensitivity to noise. Consequently, a surrogate model with flexible, non-interpolating, and parsimonious characteristics is required to overcome real-world noisy black-box functions with only a subset of important variables. One such surrogate model is, Multivariate Adaptive Regression Splines (MARS) which was initially developed by Friedman. In this study, we propose a modified version of MARS, Tree Knot MARS (TK-MARS), to improve the application of MARS within the surrogate optimization context. TK-MARS is able to identify the peaks and valleys for optimization using a classification and regression tree partitioning method. Furthermore, we develop a smart replication strategy based on hypothesis testing. The Smart-Replication approach identifies the promising points to replicate and the number of replications for each of them.

Table of Contents

Chapter 1

Introduction

1.1 Black-Box Systems

Simulators can be used to study, and potentially optimize many real-world complex systems such as air quality [1], green building [2], hospital emergency room [3], investment portfolio [3], vehicle crash [4], and agricultural crop ideotype [5] simulation models that are called black-box systems. Black-box systems are complex systems of inputs and outputs with entirely unavailable information about the underlying behavior and model. Hence, there is no mathematical formulation for these types of functions, and as a result, the derivatives are unavailable. Consequently, the common derivative-based optimization techniques cannot be employed.

In this research, our focus is on solving an optimization problem of the following form:

$$\min f(x) \tag{1.1}$$

$$s.t.$$

$$a \leq x \leq b, \forall x \in \mathbb{R}^d, \tag{1.2}$$

where $f(x)$ is the black-box function, and the goal is to obtain a globally optimal solution of $f(x)$ in the feasible input space of D, where $D = \{x \in \mathbb{R}^d : a \leq x_j \leq b, \forall j = 1, \cdots, d\}$.

The black-box optimization problem formulation is like any other optimization problem with an objective function and constraints defining the feasible region. However, the difference is in the objective function, $f(x)$, which is totally unknown and usually has a complex underlying behavior.

Evaluting the black-box function $f(x)$ involves expensive experiments that may either be computer simulators like energy simulation tools or actual experiments like crash simulators. Reaching an optimal solution of a high-dimensional expensive black-box function in a limited number of function evaluations is the main challenge of the problem. Therefore, finding a best sampled mean solution (BSMS) in a fixed number of function evaluations (which usually depends upon a given budget or time constraint) and measuring the performance of the algorithm is a matter of concern.

Since these simulators are often too computationally expensive to be embedded directly within a global optimization method, a more practical approach em-

ploys surrogate models that are computationally less expensive to evaluate. One of the existing well-known derivative free techniques for solving black-box optimization is surrogate optimization.

A surrogate model is a mathematical approximation of the relationship between input and output variables. Several types of statistical models including Kriging [6], Radial Basis Functions (RBF) [7], Regression Trees [8], Multivariate Adaptive Regression Splines (MARS) [9], Artificial Neural Networks [10], Support Vector Regression (SVR) [11], etc. are distinguished as surrogate models.

Surrogate optimization requires careful selection of simulator runs so as to simultaneously improve the surrogate model and gain more information on potential optimum in each iteration.

1.2 Motivation

The purpose of this study is to develop an algorithm that finds an optimum input vector of a high-dimensional black-box system that includes uncertainty and has some unimportant input variables in few experiments (function evaluations).

Historically, surrogate optimization methods assume the complex black-box system involves no uncertainty in performance and the set of important decision variables is known *a priori*. In the real world, both of these assumptions are often violated. This research introduces a new surrogate optimization paradigm to address uncertainty in system performance and feature selection for important decision variables. In real simulations a large percentage of variables are unimpor-

tant [12], and there are uncertainties associated with the black-box system [13]. Every variable being significant is the weakest part of the literature. We need a new surrogate optimization method to handle this paradigm shift.

Interpolating models, such as Radial Basis Function (RBF) and Kriging, are the most common techniques applied in surrogate optimization literature, which cannot handle noise inherently and assume every input is significant.

1.3 Research Questions

In this research we address the following primary research questions:

- How can we handle unimportant inputs (decision variables of the black-box function) in surrogate optimization?
- How can we handle the uncertainties associated with the black-box functions in surrogate optimization?

1.4 Contributions

To answer the research questions, we develop a new surrogate optimization approach, Tree-Knot Multivariate Adaptive Regression Splines (TK-MARS).

Existing statistical models are mainly designed to fit the data, precisely. For surrogate optimization, we need a specifically designed statistical model to determine the overall structure of the underlying function, where a potential optimum

lies, in the fewest number of function evaluations. The goal is finding an optimal solution, not modeling the black-box function.

MARS has classically been used for prediction, not for surrogate optimization. As a surrogate modeling method, we employ MARS for its feature selection ability and its robustness toward uncertainty. MARS can handle the fact that there are some unimportant variables as opposed to conventional interpolating-based surrogate models such as Radial Basis Function (RBF) that fail in handling this issue. The other advantage of MARS as the surrogate model is its robustness to the false structures caused by noise for small samples in high dimensions. The MARS forward selection and backward elimination procedures make the model parsimonious based on the important variables. Since MARS is a regression-based approach it is able to cancel out the noise effect by passing among the points and minimizing errors.

In this study, a new version of Multivariate Adaptive Regression Splines, Tree-Knot MARS (TK-MARS), is developed specifically for surrogate optimization . We modified the MARS knot selection algorithm by using Classification And Regression Trees (CART). TK-MARS focuses on improving the accuracy in the promising regions. It is able to identify peaks and valleys where potential optimum solutions lie. The proposed eligible knot selection technique improves the flexibility of MARS as well as adds promising candidate points to the data set simultaneously.

Compared to existing surrogate optimization approaches, the presented approach can handle simulators with a large number of inputs and performance

uncertainties. TK-MARS is able to identify important inputs of the black-box function. Centroids from TK-MARS are used to develop Dynamic Pool Generation.

In addition, to improve the selection of simulator runs in each iteration, we employ a Pareto approach to balance exploration and exploitation. An Exploration and Exploitation Pareto Approach, EEPA [14] finds the best candidate points to be added in each iteration. Further, we modify EEPA to incorporate TK-MARS and a standardized Cosine similarity metric.

A new measure of performance is proposed to evaluate the different approaches. Maximal True Function Area Under the Curve (MTFAUC) is a proper metric for stochastic black-box functions in which the algorithm BSMS convergence pattern is unstable in early iterations. MTFAUC quantifies the stability and robustness of the algorithm, simultaneously.

In the end, different replication strategies are proposed to handle the uncertainties associated with black-box functions.

1.5 Organization

Chapter 2, introduces surrogate optimization algorithms and related work in the literature to highlight the gaps that are not investigated properly. It provides a background on MARS and RBF methods as well as the candidate selection approach that is applied in our study. Chapter 3 describes the contributions of our study to the surrogate optimization algorithm and concentrates on the proposed

TK-MARS method. The concept of unimportant variables in the context of surrogate optimization is explained in this section. Furthermore, the proposed replication strategies to handle the uncertainties associated with the black-box systems are presented as well as a new measure of performance. Chapter 4 demonstrates the results of the performance of TK-MARS and also the performance of the proposed approaches in deterministic and stochastic cases. Finally, Chapter 5 lists potential future directions of this study.

Chapter 2

Background on Surrogate Optimization

Surrogate optimization represents a framework in a class of derivate-free optimization methodologies, which has been applied for expensive black-box functions. It is a sequential adaptive approach to discover the solution space, and find a global optimum. Surrogate models are response surface models, which estimate the underlying behavior of black-box functions. It is worth mentioning that, unlike black-box functions, the surrogate models are inexpensive functions to execute. We refer to the output of the black-box function as the *actual objective value*.

In this chapter, we present a background on the surrogate optimization framework and details of different steps. Figure 2.1 shows the general overview of a generic surrogate optimization algorithm.

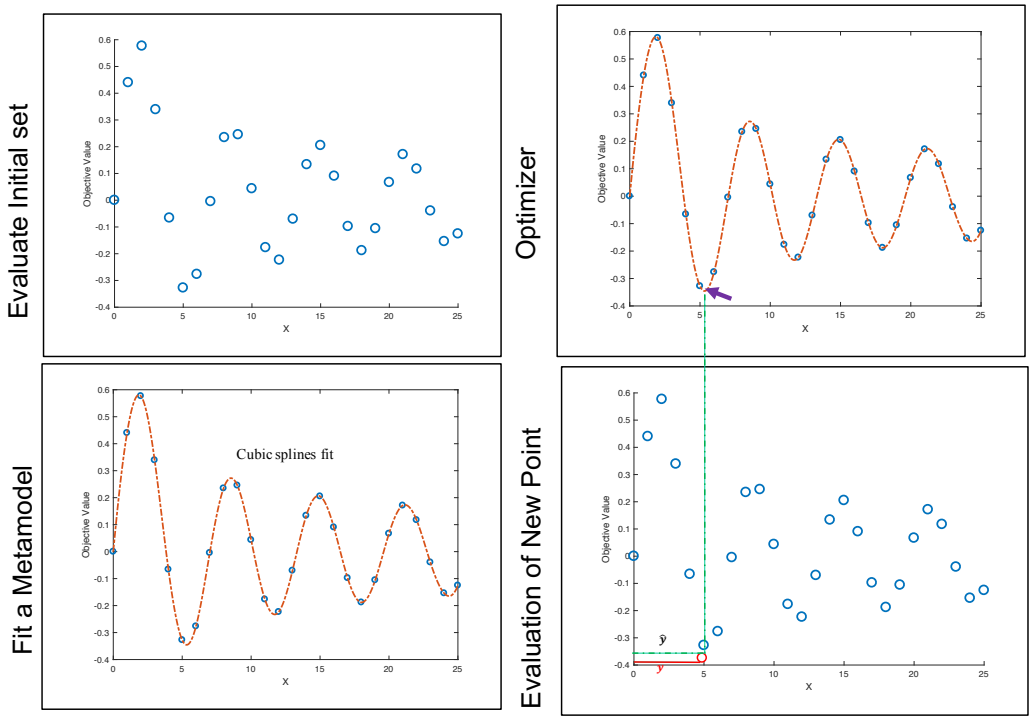


Figure 2.1: Surrogate Optimization Framework Illustration

There are specific steps in all surrogate optimization approaches. First of all, a small set of data points are initialized to be evaluated by the black-box function. Next is the metamodeling step, which fits a response surface model to the initial data set; in Figure 2.1 we use a cubic splines interpolation method. The following step is to apply an optimizer, which determines new candidate points by solving an auxiliary optimization problem.

Ideally, surrogate optimization attempts to find a global optimum of the surrogate model and evaluate it with the black-box function to get the actual objective value for it and add the newly evaluated point to the initial data set. Algorithm 1 is an overview of a generic surrogate optimization algorithm for black-box func-

tions.

The algorithm iterates until a stopping criterion is satisfied. The stopping criteria can either be a maximum number of costly black-box function evaluations or the expected improvement of the best sampled mean solution (BSMS).

Algorithm 1 Surrogate Optimization

1: Sample initial design space $I = \{x^i \in \mathbb{R}^d \mid \forall i = 1, \cdots, N\}$
2: Evaluate initial data set I, $f(x^i), \forall x^i \in I$
3: **while** Termination criteria not satisfied **do**
4: Construct a surrogate model on $|I|$ evaluated points, $\hat{f}$
5: Optimizer (search for new candidate points based upon $\hat{f}$ and I), P
6: Evaluate selected candidate points, P, $f(x^k), \forall x^k \in P$
7: Update the collection of already evaluated data points, $I = I \cup P$
8: **end while**
9: Return the BSMS, $x \in \operatorname{argmin}_{x \in I} f(x)$

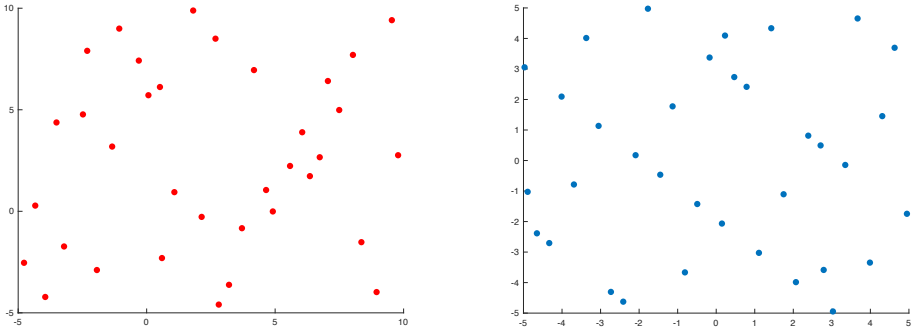
The remainder of this chapter, describes surrogate optimization steps in detail. We review different metamodeling techniques and different optimizers that are used in the literature.

2.1 Initialization

The problem description is presented in Section 1.1. The feasible solution space is defined as D where $D = \{x \in \mathbb{R}^d | a \leq x_j \leq b, \forall j = 1, \cdots, d\}$. Due to the high cost of function evaluations of $f(x)$, the algorithm samples a limited number of feasible initial points using design of experiments methods (DOE). Space filling sequences, including Latin Hypercube Design (LHD), Sobol [15,16],

and Orthogonal Arrays (OA) [17], represent the solution space well. The size of the initial data set is $|I| \geq d + 1$.

Figures 2.2a and 2.2b are schematics of 35 sample points chosen by Latin hypercube sampling (a) and Sobol (b) for a 2-D problem.



(a) LHD (b) Sobol

Figure 2.2: Design of Experiment methods for initialization

2.2 Function Evaluation

After choosing the initial candidate points, the algorithm needs to find the actual objective values by executing the expensive black-box function. Later, every sample point that is selected by the optimizer is evaluated, sequentially. We want to emphasis that one of the goals is to evaluate as few sample points as possible to evaluate.

2.3 Surrogate Model Construction

Choosing an appropriate surrogate model highly depends upon the performance of different metamodeling techniques under different circumstances. There are several metamodeling techniques that are used in the surrogate optimization context. Surrogate models are classified into either interpolating (kriging [6], RBF [7], etc.) or non-interpolating (polynomial regression models [18], multivariate adaptive regression spline [9], etc.).

2.3.1 Non-Interpolating Models

Non-interpolating surrogate models are mainly based on the least squared error of some predetermined functional form. Unlike interpolating surrogate models, they do not necessarily pass through all the data points to capture the exact behavior of the function. In terms of bias-variance trade off, non-interpolating surrogate models may have higher bias and lower variance than interpolating surrogates. Therefore, in highly noisy systems, non-interpolating models are preferred in order to avoid oscillations of the fitted surface.

2.3.1.1 Polynomial Regression Models

Polynomial regression models have been used for decades in black-box optimization [19–22]. Regression models are based on the relationships between factors that are suspected to have an influence on the response. The mathematical formulation is commonly expressed as $y = \hat{f}(x) + \epsilon$, where x is a vector of one or

more factors, and ϵ is a random error. Different regression models are introduced regarding their functional form. In polynomial regression, the response variable y is modeled as an m^{th} degree polynomial in x, and is of the form given by Equation 2.1. This model is able to smooth noisy functions.

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_m x^m + \epsilon, \tag{2.1}$$

where β_0 is the intercept, and β_j is the coefficient for the j^{th} dependent variable $\forall j = 1, \cdots, m$.

2.3.1.2 Multivariate Adaptive Regression Splines (MARS)

Friedman [9] develops the original version of MARS. MARS is a non-parametric regression-based surrogate model. The advantages of MARS over other statistical models, such as linear regression models lie in its ability to handle curvature in high-dimensional space and produce easier-to-interpret models. Furthermore, the MARS algorithm includes forward selection and backward elimination procedures. In the forward selection procedure, MARS adds basis functions in pairs for different dimensions that give the maximum reduction in the sum of squares error, until the maximum number of basis functions is reached. The backward elimination procedure prevents overfitting by deleting the least effective term at each step.

Therefore, MARS is intended to be parsimonious. Basically, it has a built-in dimension reduction technique, which is very beneficial for black-box optimiza-

tion where there is no prior knowledge of input variables and their effect on the response. As a result, it keeps the most significant variables in the final model.

The structure of MARS model is based on basis functions. The MARS algorithm uses splines to construct a piecewise continuous function to model the dependent variable. The basis functions are either univariate truncated linear functions or the product of different truncated linear functions for interaction terms. The truncated linear functions also called hinge functions are of the form $b^+(x;k) = [x-k]_+$ or $b^-(x;k) = [k-x]_+$, where $[q]_+ = \max\{0, q\}$, x is a single independent variable, and k is the corresponding univariate knot, where the approximation bends to capture curvature of the black-box function. A knot is the location of an intersection between two splines and therefore is an important concept associated with MARS. The eligible knot locations are selected from the set of training points. Each dimension value of the training data set is an eligible knot for the corresponding independent variable $\{x^1, \ldots, x^N\}$.

The MARS approximation has the form:

$$\hat{f}(x, \beta) = \beta_0 + \sum_{m=1}^{M_{max}} \beta_m B_m(x), \tag{2.2}$$

where x is an n-dimensional vector of input variables, β_0 is the intercept coefficient, M_{max} is the maximum number of linearly independent basis functions, β_m is the coefficient for the mth basis function, and $B_m(x)$ is a basis function that is either univariate or multivariate interaction terms. The interaction terms are

presented in the form of:

$$B_m(x) = \prod_{l=1}^{L_m} [s_{ml}(x_{v(m,l)} - k_{ml})]_+, \tag{2.3}$$

where L_m is the number of interaction terms in the mth basis function, $x_{v(m,l)}$ is the vth dependent variable corresponding to the lth truncated linear function in the mth basis function, and k_{ml} is the knot value corresponding to $x_{v(m,l)}$. The value s_{ml} is the direction that the truncated linear basis function can take, either +1 or -1.

A good value for the maximum number of basis function parameter of MARS, M_{max}, is usually obtained through trial and error. However, a basic modeling concept is that as sample size increases, the model complexity can be increased. Using a small M_{max} when the number of data points are limited, is reasonable. Sahu [23] recommends $M_{max} = \left[\frac{2n+c}{2+c}\right]$, where c is a penalty, which the default is 3.

The maximum number of knots, the minimum number of observations between knots, and the highest order of interaction terms should be predetermined in the original MARS algorithm. The MARS approximation procedure employs a forward selection and a backward elimination procedure to choose a small subset of appropriate basis functions. The basis functions are defined by the eligible knot locations that minimize the lack of fitness. Equation 2.4 shows the LOF criterion used by MARS for parameter optimization.

$$LOF = \sum_{i=1}^{N} [y^i - \hat{f}(x^i, \beta)]^2, \tag{2.4}$$

where $\hat{f}(x, \beta)$, is the approximation of MARS as in Equation 2.2 and Equation 2.3. To find the best set of parameters β, MARS minimizes the LOF function.

In the forward selection procedure, MARS starts with only the intercept term, and repeatedly adds pairs of basis functions, which minimize the LOF the most, to the model. The forward selection procedure is a greedy approach may add too many basis functions that may lead to overfitting. In order to prevent overfitting, the backward elimination procedure is designed to eliminate less effective basis functions. The number and the location of knots are important parameters of the MARS algorithm. In the conventional MARS, all data points are considered in order to find a subset of promising knot locations.

Calculating the averaged squared residuals over all possible locations for knots is computationally difficult, especially when the number of eligible knots and basis functions is unknown in advance. Highly noisy data can lead to high local variances in function estimation. In addition, when there are a large number of eligible knots to be selected, a multicollinearity issue occurs between the basis functions with knots that are close to each other.

To solve local variance and multicollinearity issues, eligible knot selection techniques are developed. One of the simplest methods considers evenly spaced eligible knot locations chosen to be equally spaced in a certain interval within the range of the input data [24–27]. Evenly-spaced knots may not capture true

patterns in the data, and a small number of knots may result in underfitting. Friedman [9] proposes the minimum span (MinSpan) approach to minimize the local variability. In the MinSpan approach, for each independent variable, a local search around its current knot location is designed to reduce the number of eligible knot locations. Miyata [28] presents a simulated annealing approach to choose eligible knot locations. Recently Koc [29] develops a mapping strategy by transforming the original data into a network of nodes through a nonlinear mapping. Each node has a specific topological position in the lattice and is represented by a $p + 1$-dimensional weight vector. The nodes in the mapped network are basically a reference for choosing the new candidate knots. The size and structure of the grid are preset in advance. The nodes are located with equal space on the grid. The grid size and a threshold value are self-organized mapping parameters, which have an impact on the accuracy and time efficiency of the model. Decreasing the grid size and increasing the threshold value lead to fewer candidate knot locations and decrease CPU time and model accuracy. Hence, the mapping parameters need to be optimized beforehand.

Tsai and Chen [30] propose a robust version of MARS, which tends to choose lower order interaction terms. Robust MARS decreases the sensitivity of MARS to extreme points, preferring lower order interaction terms to higher order terms by weighting the lower order terms. Univariate basis functions and interaction basis functions could be highly correlated, so robust MARS chooses the lower interaction terms in the range of a tolerance, to handle the multicollinearity issue. Robust MARS is only relevant if interactions are allowed in the model.

Instead of M_{max}, an alternative stopping rule, Automatic Stopping Rule (ASR), criteria based on the coefficient of determination, R^2, is implemented to select the appropriate M_{max}. The M_{max} should be pre-specified; however, an improper value of M_{max} leads to underfitting or overfitting issues. ASR provides an approach to use the optimum number of basis functions in the model to avoid unnecessary computations of a large M_{max}.

Crino [31] presents a combination of MARS and Response Surface Methodology (RSM). MARS is employed to select the initial metamodel. Regarding the variable reduction ability of MARS, the underlying function can be approximated by using an RSM with fewer variables. The proposed method is applied to low-dimensional test functions. Costas [32] employs MARS for a real-world finite element model and compares it with Kriging, quadratic, and cubic polynomials. MARS outperforms other methods due to its ability to model slope discontinuities and to filter almost all numerical noise.

2.3.2 Interpolating Models

Applying interpolating surrogate models are common in surrogate optimization literature. Interpolating models pass through all of the data points and consider the exact information of the function value in the approximation. Although this can help to determine the underlying behavior of the black-box function more accurately, the quality of model generalization depends on the size of the data set I.

2.3.2.1 Kriging

Kriging is an interpolating surrogate model that has been applied in black-box optimization, widely. Kriging provides an estimation of a black-box function minimizing squared error [6]. It has been typically utilized in surrogate optimization literature for low-dimensional problems since its computational time for large-scale problems makes it inefficient [33, 34]. Kriging models consist of two components. The first component is a model that captures the trend in the data, whereas the second component measures the deviation between the trend model and the true function value. The Kriging model structure is given by the following equation.

$$\hat{f}(x) = \sum_{i=1}^{n} \beta^i y^i + Z(x), \tag{2.5}$$

where $Z(x)$ is a correlation function and is assumed to be an uncertainty estimate. β^i is the coefficient for the response value of observation i. The most common correlation function is the Gaussian correlation function [33].

2.3.2.2 Radial Basis Function

RBF is another common type of surrogate model [7]. Assuming n distinct already evaluated points, $x^1, x^2, \cdots, x^n \in \mathbb{R}^d$, the RBF interpolant is of the form, $\hat{f}(x) = \sum_{i=1}^{n} \lambda^i B(||x - x^i||) + p(x), x \in \mathbb{R}^d$ where $||.||$ refers to the L_2 norm, and $B(x)$ is the basis function of the following forms:

- A *linear* basis function is $B(r) = r$

- A *cubic* basis function is $B(r) = r^3$
- A *thin plate spline* basis function is $B(r) = r^2 log(r)$
- A *Gaussian* basis function is $B(r) = e^{\frac{-r^2}{\omega^2}}$ where ω is a positive constant.
- A *multi-quadric* basis function is $B(r) = \sqrt{r^2 + \omega^2}$
- An *inverse multi-quadric* basis function is $B(r) = \frac{1}{\sqrt{r^2+\omega^2}}$

For multiquadric and Gaussian basis functions, a unique interpolant is guaranteed when solving the RBF equation. However, for cubic and thin plate spline, the matrix of basis values might be singular. Rocha [35] adds a low degree polynomial, $p(x)$, to RBF interpolations to avoid singularity.

The most common types of basis functions are Multiquadric (MQ), Gaussian (G), Cubic (C), and Thin Plate Splines (TPS). The multiquadric version of RBF is continuously differentiable. The shape parameter ω and the number of points affect the accuracy and stability of RBF (a larger shape parameter makes the function flatter). It is worth mentioning that MQ guarantees non-singularity [36].

2.4 Optimizer: Candidate Selection

The optimizer procedure in surrogate optimization is crucial due to the expensive function evaluations of black-box systems. The goal is to select the next sample point so that it finds an optimum solution in a few function evaluations. Exploration involves discovering unexplored regions of interest, and exploitation digs into the identified regions of interest where the optimum solution probably lies. As a result, a balance between exploration and exploitation must be satisfied for

candidate selection.

To determine the next sample point, different strategies are proposed in the literature. Moore [21] introduces a new candidate selection approach for black-box optimization. The proposed algorithm, $Q-2$ identifies the neighborhood of the optimum solution by capturing the size and shape of the zone of possible optimum locations. After evaluating each of the data points, it finds the point that has the worst prediction value and cuts the space using a half-plane $ROI_2 = ROI_1 \cap \{x \mid (x - x_{k(1)} \cdot d_1 \geq 0\}$ using the direction of the steepest gradient of the estimated y. $x_{k(1)}$ is the point with the worst predicted value (using quadratic regression) and $d_1 = \bigtriangledown \hat{y}$. After finding the ROI, x_{opt} is identified. The next sample point is either: 1) the x_{opt} location, (2) a random point within the ROI, (3) the point in ROI that is predicted to reduce the uncertainty the most, (4) the point in the ROI that keeps the regression as orthogonal as possible, or (5) the point in the ROI as far away from previously evaluated points in or out of the ROI. $Q-2$ is not functional for larger than 10 dimensions, and it can only find local optima.

Regis [37] proposes a candidate selection method for costly function evaluation minimizing the response surface model subject to distance constraints. The method is tested on low-dimensional test problems, and the disadvantage of the method is adding a single solution each time. Gautman [38] solves an auxiliary optimization problem to find the candidate points, as well.

Regis [39–41] applies a random sampling strategy for candidate selection. The proposed approach is based on a random pool that is dynamically generated in

each iteration by adding normal perturbations to the current best point. The feasibility of the candidate point is checked with the number of predicted constraint violations. A weighting approach on the distance measure and response surface is applied to satisfy the exploration-exploitation balance of the next candidate point. The proposed algorithm focuses more on finding good local solutions from a given feasible starting point, and it reduces exploration (localized search). In addition, a single candidate point is selected in each iteration.

Regis [42,43], and Dong [44] propose an evolutionary approach for generating candidate points. A large number of trial offspring are generated by using Gaussian mutations in each iteration. RBF is used for objective and constraint functions to identify the most promising feasible offspring that has the best-predicted value and least constraint violations. The selected offspring becomes the actual candidate point that is evaluated by the true expensive black-box function. A Pareto approach is employed to evaluate a set of offspring in each iteration. The proposed method in [43] is tested on low-dimensional test problems, and it adds one single sample point in each iteration. Both [42, 43] are evolutionary-based methods, and like other evolutionary approaches, there is no guarantee in convergence to the optimal solution. The approach is sensitive to prediction accuracy (quality of the approximation).

Jones [45] and Knowles [46] apply Expected Improvement (EI) in the objective function to select the next candidate point. The point with the highest EI is selected. EI is maximized to search the feasible space globally. An Efficient Global Optimization (EGO) algorithm balances exploitation and exploration by

minimizing prediction cost and error. Knowles [46] extends the EGO algorithm for multiobjective optimization problems. ParEGO embeds a genetic algorithm to visit the next candidate point, which maximizes EI. Dong presents a Kriging-based balance between exploitation and exploration in [44].

A Pareto non-dominated sorting technique has been recently taken into consideration to select multiple sample points at a time [14, 47, 48]. Dickson [14] presents the idea of using a Pareto frontier for the exploration-exploitation trade-off for the first time. EEPA is a sampling technique for surrogate optimization, which combines an exploration metric (a distance metric) with an exploitation metric (predicted objective value). Dickson [14] designs a fixed random pool to search for the Pareto frontier.

Krityakierne [48] generates a random pool repeatedly in each iteration by perturbations around some center points. The centers are the points, which have the best objective values and maximum distance from the already evaluated points to balance between exploration and exploitation. To find many points for simulataneous evaluation, parallel processors are considered. Consequently, using a Pareto approach, a set of centers are identified for each processor, which represents the center of the random pool to be generated. Consequently, parallel pools are evaluated to find new samples for the next generation.

Bischl [47] develops a Pareto frontier approach as a multi-objective solution to trade off between exploration and exploitation criteria. Different objectives are employed to find the Pareto set of sample points by using evolutionary algorithms (EA). These criteria include mean model prediction, standard error (model un-

certainty), Expected Improvement, distance to the nearest neighbor in the current population, and distance to the nearest better neighbor.

2.4.1 EEPA: Exploration-Exploitation Pareto Approach

EEPA [14] is an exploration-exploitation algorithm that creates a Pareto frontier on the estimated function value from the surrogate model, as one dimension, and the distance of the candidate points from the already evaluated points, as the second dimension. The first dimension exploits the interesting regions around the minimum function value area, and the second dimension explores the undiscovered regions. The exploration metric is $\delta(x) = \min_{\tilde{x} \in I} \|x - \tilde{x}\|$, and the exploitation metric is the fitted function value $\hat{f}(x)$. The first metric needs to be maximized, while the second should be minimized. Hence the Pareto set is the set of non-dominated solutions given by $F = \{x \in R \mid \nexists \tilde{x} \in R, \hat{f}(\tilde{x}) \leq \hat{f}(x), \delta(\tilde{x}) \geq \delta(x)\}$, where R is a fixed random pool of sample points.

Algorithm 2 shows EEPA for surrogate optimization. It starts with designing an initial data set I and a set of random points R from which candidate points will be selected. In step 3 the computer model is executed to obtain the actual function values. The next step is to fit a metamodel to the data set. In steps 5 and 6, the two criteria, distance and the predicted objective value, are calculated to find the non-dominated Pareto set in step 7. There might be candidate points on the Pareto set that are close to each other. The final step is to find the winning candidate points from the Pareto set to add to the set I using *maximin* exploration technique.

Dickson [14] shows that EEPA outperforms pure exploration and exploitation

Algorithm 2 EEPA, [14]

I is the initial set, R is the random data set
For each $x \in I$, determine $f(x)$
while Termination criteria is not satisfied **do**
  Construct a metamodel $\hat{f}$ using $(x, f(x)), x \in I$
  For each $x \in R$, determine $\hat{f}(x)$
  For each $x \in R$, determine $\delta(x) = \min_{\tilde{x} \in I} ||x - \tilde{x}||$
  Determine $F = \{x \in R \mid \nexists \tilde{x} \in R, \hat{f}(\tilde{x}) \leq \hat{f}(x), \delta(\tilde{x}) \geq \delta(x)\}$
  Set $P = \emptyset; k' = 1$
  Determine $x' \in \arg\min\{\hat{f}(x) | x \in F\}$
  $P = P \cup x'$
  while $k' \leq K'$ **and** $P \subset F$ **do**
    For each $x \in F$, determine $\delta'(x) = \min_{\tilde{x} \in A \cup P} ||x - \tilde{x}||$
    Determine $P = P \cup \{x\}, x \in \arg\max\{\delta'(x) | x \in P\}$
    $k' = k' + 1$
  end while
  For each $x \in P$, determine $f(x)$
  $I = I \cup P$
  $R = R \setminus P$
end while
Return $x^* \in \arg\min\{f(x) | x \in A\}$

methods. However, EEPA and the quality of BSMS found by EEPA depends on the fixed random data set R.

2.5 Measure of Performance

To evaluate the performance of surrogate optimization algorithms either on a test function or a real black-box simulation model, several performance measures can be found in literature. Gutmann [38] and Jakobsson [49] employ the relative error, Equation 2.5, for deterministic functions.

$$e^i = \frac{|f(x^{oi}) - f^*|}{|f^*|}, \forall i = 1, \cdots, |I|. \tag{2.6}$$

where f^* denotes the global minimum of the function value and x^{oi} is the minimum proposed by the algorithm, BSMS, after i function evaluations. The algorithms are terminated when the relative error after i function evaluations e^i reaches below a threshold e.g. 1%, or when the number of function evaluations exceeds some fixed number (representing the available budget), whichever comes first. Besides, a graphical illustration of the differences between the performance of the algorithms is also a valuable tool that has been utilized in the literature to show convergence and the capability of the algorithms.

There is little research that consider uncertainties related to black-box systems [49,50]. For non-deterministic cases, a performance measure of convergence that is applied is presented in Equation 2.7.

$$G^i = \frac{f_{x^{in}} - f(x^{oi})}{f_{x^{in}} - f^*}, \tag{2.7}$$

where x^{oi} is the minimum proposed by the algorithm, BSMS, after i function evaluations, and f is the true function. The point x_{in} is the median initial point $x_{in} \in \arg\text{median}_{x \in I} f(x)$, where I is the set of initial points. The measure is constructed such that $G_i \leq 1$ for all i. Reaching $G_i = 1$ means that the global minimum is found.

2.6 Research Gap

Above, we explain surrogate optimization algorithms and related research in the literature. To summarize, there are different surrogate metamodeling techniques that can be classified as interpolating or non-interpolating. Figure 2.3 shows different metamodels in the literature in chronological order. Regression and MARS as non-interpolating and RBF and Kriging as interpolating surrogates are the most common metamodeling techniques. Among these, [14, 21, 31, 32] use non-interpolating models such as MARS and polynomial regression. The rest are focused on interpolating surrogates considering all the variables important. Every variable being important is the weakest part of the surrogate optimization literature. Crino [31] shows MARS is able to screen variables and reduce dimensions. Using the parsimonious nature of MARS, we develop TK-MARS that is able to identify the important variables.

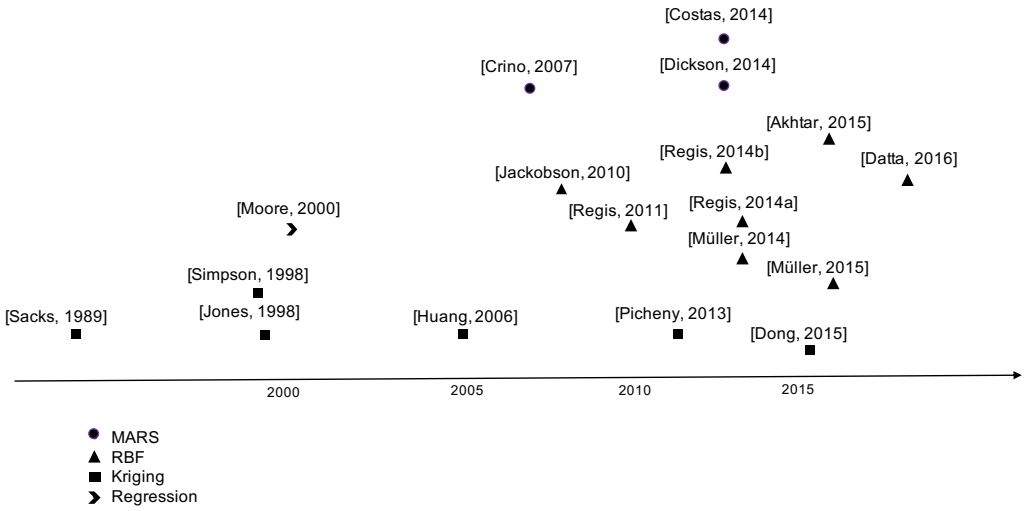


Figure 2.3: Metamodeling Literature

From all the research displayed in Figure 2.3, [13, 31, 32, 49, 50] consider uncertainty components in metamodeling. Kriging and RBF, the two most common surrogate models, do not inherently handle uncertainty, and some modifications are required while MARS and regression require no revision to handle uncertainty.

Huang [50] develops a Kriging-based surrogate optimization approach and applies it to low-dimensional test problems including low-level noise. The cost of fitting Kriging increases by the number of samples as it leads to impractical higher dimensional problems. The proposed method under higher levels of noise for very bumpy objective functions needs further investigation.

Picheny [51] adds Gaussian noise with a fixed independent variance from the response and performs it on low-dimensional optimization test problems. The results show the relative poor modeling performance of Kriging. A large part of the variability that cannot be explained by the model is due to the observation noise during optimization. Further, analysis of variance of modeling parameters is presented to show the significance of the modeling parameters, such as initial design, different infill criteria, the noise level of test functions, and covariance kernel. Jakabsson [49] and Picheny [13] apply RBF and Kriging based surrogates, respectively. However, the proposed methods are performed only on low-dimensional test problems that include low levels of uncertainty.

[13, 31, 32, 49, 50] propose metamodels that are able to handle uncertainty, even though they do not study the effect of the noise. Costas [32] shows MARS is preferable in real-world applications due to slope discontinuities and uncertainties. We show TK-MARS is able to handle uncertainties associated with the black-box

systems.

Figure 2.4 represents some of the surrogate optimization research that are mainly focused on developing an efficient optimizer. We categorize them based on the different candidate selection approaches.

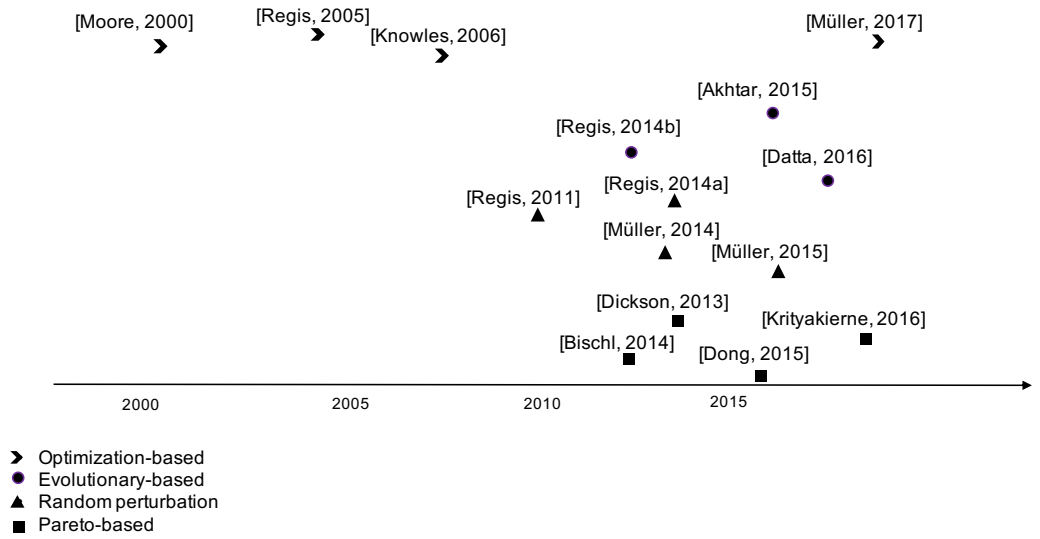


Figure 2.4: Surrogate Optimization Literature

The Pareto-based candidate selection approach has been recently taken into consideration to select multiple sample points at a time [14, 44, 47, 48]. We apply EEPA [14] in this research and modify it to incorporate TK-MARS and a standardized Cosine metric.

Chapter 3

TK-MARS-based Surrogate Optimization Approach for Deterministic and Stochastic Black-Box Functions

In Chapter 2, we describe the generic surrogate optimization procedure, alternative surrogate models, and candidate search methodologies that are common in the literature. In this section, we present the proposed surrogate model that is specifically designed for surrogate optimization of black-box functions. Furthermore, a centroid-based dynamic pool generation approach is developed to improve the candidate selection procedure. In this research, we propose an alternative approach to identify subregions of interest, which nicely surround promising points

for subsequent function evaluations. We introduce uncertainty associated with the black-box functions and propose effective strategies to handle it in the surrogate optimization procedure.

In Section 3.1, we describe details of the surrogate model that is specifically designed for surrogate optimization of expensive black-box functions. In Section 3.2, we discuss the concept of unimportant variables in surrogate optimization. Further, we propose a new convergence approach in Section 3.3. To handle uncertainties associated with black-box functions, we propose different approaches in Section 3.6.

Next, in Chapter 4, we present computational experiments on problem and algorithm parameters to study the performance of the proposed approaches. We also study the impact of different parameter values on the performance.

3.1 Tree-based Eligible Knot Selection Approach: TK-MARS

We introduce Multivariate Adaptive Regression Splines (MARS) in Section 2.3 as a non-interpolating surrogate model. In this section, we develop a modified version of MARS for surrogate optimization. As we discuss, the MARS structure is based on basis functions, which include knots where the approximation bends.

In MARS, all the distinct data points are considered as eligible knot locations. Adding more data points increases the size of eligible knot locations. As the number of eligible knots increases, MARS interpolates the data points and loses its

flexibility. Interpolating may cause difficulties for the surrogate optimization procedure in the presence of uncertainty. Interpolating a limited data set early when there is not enough information collected leads to a false estimation of function behavior that causes several local optima. The fact that MARS does not interpolate but finds the general structure of the function is the major reason for applying MARS to surrogate optimization. Considering too many eligible knots for MARS results in including all basis functions, which yields a poor approximation and overfitting. By choosing a subset of eligible knot locations for the first set of basis functions and slowly allowing MARS to add more knots as the set of training points increases, the complexity of MARS is restricted, and the flexibility is escalated. For highly noisy data, considering all the data points as eligible knot locations leads to high local variance in function estimation [29]. Further, when a large number of eligible knots are selected, multicollinearity can occur between basis functions with knots that are close to each other. There are few approaches in the literature that are proposed for eligible knot selection as we discuss in Section 2.3.1.2.

The idea behind the proposed modified MARS is to fit MARS accurately around potential optimum points. The proposed approach considers identifying the new eligible knot locations plus sampling the new candidate points, simultaneously. Along these lines, appropriate eligible knot locations are identified, and the approximation model is constructed gradually as more data are collected. Therefore, there is no need for many basis function evaluations and costly function evaluations.

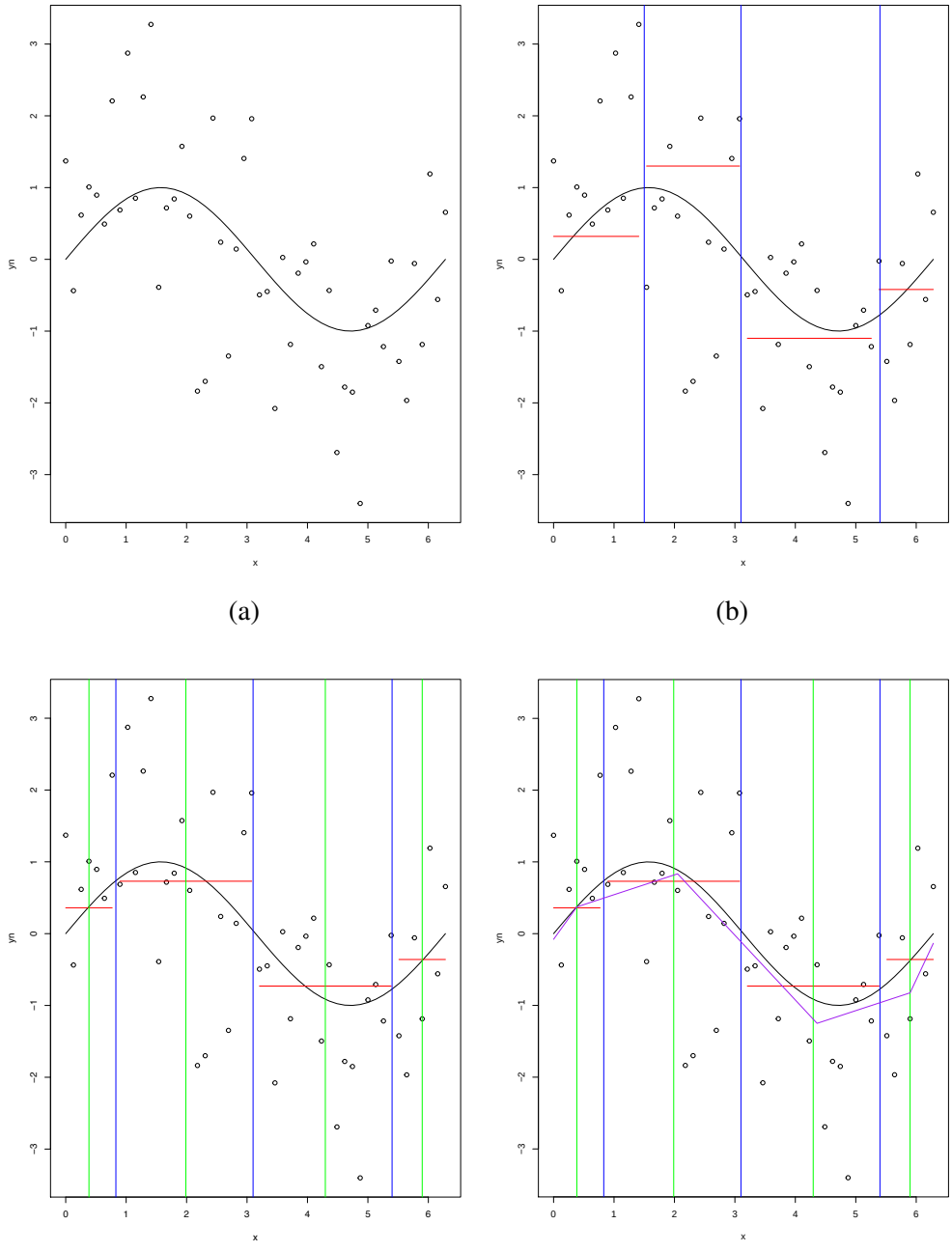
A MARS-based surrogate optimization approach is proposed in this study. The initial data points are generated by a Design of Experiments method (DOE). The size of the initial data set is limited in black-box optimization due to the high cost of function evaluations. All the initial data points are evaluated by the black-box function to find the actual function values. The proposed approach, which is called Tree-Knot MARS or briefly TK-MARS, captures the structure of the function in the solution space using a partitioning technique. Classification and Regression Tree, CART [8] (CART) is a binary decision tree, which splits the set of data points on the input variables until a suitable tree is constructed. As the data set is updated dynamically over time, more information from the black-box function structure is available. The partitions discover the structure of the unknown function over time.

Before going to the details of the TK-MARS procedure, consider the following high-level example of TK-MARS. Suppose we have a noisy data set (adding a random error to the function value) for $f(x) = \sin(x) + \epsilon$, where ϵ is a noise term with a mean of 0, as in Figure 3.1a. Consequently, $E[f(x)] = \sin(x)$. CART splits the existing data points into four partitions, as in Figure 3.1b. Note that CART is capable of identifying the structure of the underlying $\sin$ function. The vertical blue lines show the partition borders, and the horizontal red lines show the estimated average function value in each partition. Next, we identify the centroid of each partition, and we select the nearest points to the centroids as eligible knot locations. The green lines are the centroids as in Figure 3.1d. We would like to highlight that the centroids are close to peaks and valleys. Since any optimization

approach looks into the peaks and valleys where optima lie, focusing on the peaks and valleys facilitates the optimization process, as well as, identifies the function structure.

The TK-MARS model construction is based on the function structure, which is obtained through the partitioning strategy. The breaking points of TK-MARS are the nearest points to the peaks (valleys) of the function (the centroids of the terminal nodes) as in Figure 3.1d. As a result, an appropriate and efficient MARS model for surrogate optimization is developed to find the overall pattern of the underlying function.

TK-MARS has a new efficient, eligible knot selection approach that is proposed for the purpose of surrogate optimization in this study. Regarding the forward-backward procedure, MARS tends to have more knots where we have more information (dense regions) by minimizing the LOF. As a result, it captures the function structure defining more knots in dense regions. Consequently, partitioning the data set in a way that captures the structure of the underlying function also finds promising eligible knot locations for MARS.



(a) (b)

(c) (d)

Figure 3.1: TK-MARS approach

Now consider the TK-MARS algorithm.

CART is a non-parametric decision tree and nonlinear predictive modeling method [8]. CART partitions the space into smaller sub-regions, recursively, so that the interactions are manageable. It is a classification method when the response variable is categorical, and in the case of having a continuous numerical response variable, CART predicts the response value in each terminal node. For regression predictive modeling problems, CART chooses binary splits by minimizing the sum of the squared error across all data points that fall within each partition. Specifically, the least squares error is given by Equation 3.1.

$$LSE = \frac{1}{N} \sum_{v \in V} \sum_{k=1}^{K_v} (y_v^k - \bar{y}_v)^2, \tag{3.1}$$

where V is the set of terminal nodes, K_v is the number of observations in node v, and $\bar{y}_v$ represents the average of the actual response values for the points in terminal v.

The advantages of using a regression tree for partitioning data are: (1) CART finds unbiased splits, (2) CART identifies the significant variables, and (3) CART identifies the interactions. For classification, CART chooses the split positions based on either the Gini index or log-likelihood function.

Partitioning the data set by using CART, we are able to prioritize the eligible knot locations. The data set is limited in the first iteration where there is not enough knowledge about eligible knot locations. Updating the initial data set by

adding more candidate points over time, the number of eligible knot locations increases. TK-MARS starts with a small set of eligible knot locations and slowly grows as the size of the data sets grows. This yields a more stable MARS model. The training data set is updated gradually. As a result, the dense regions, where we have more information about the function, are identified by partitioning, and the structure of the underlying black-box function can be recognized.

The representatives of each partition are considered as a reference point for an eligible knot location. CART partitions the data set based on the structure of the response function. Hence, there are more partitions in highly structured regions.

The peaks and valleys of the black-box function are often in the middle of the space defined by the tree logic at the terminal nodes rather than the edges since CART splits where the response values change, significantly. Hence, the centroids as calculated by Equation 3.2, are appropriate locations for the eligible knots.

$$c_j^v = \frac{1}{K_v} \sum_{k=1}^{K_v} x_j^k \quad \forall v = 1, \cdots, V, j = 1, \cdots, d, \tag{3.2}$$

where c_j^v is the centroid of terminal v for dimension j, and x_j^k is the jth dimension of observations in terminal v.

Knots in MARS are in one-dimensional space, i.e., each independent variable has its own possible values for the eligible knot locations. The centroids of terminal nodes are in multidimensional space. As a result, a transformation from multi-dimensional to one-dimensional space is required. Equation 3.3 represents

a transformation function. For each dimension, the eligible knot location is the nearest point to the centroid of terminal v in the same dimension.

$$t_j^v \in \arg \min_{k=1,\cdots,K_v} |x_j^k - c_j^v| \quad \forall j = 1, \cdots, d \tag{3.3}$$

t_j^v is the index of the observation in the initial data set that is the nearest to the centroids of each terminal nodes for each independent variable. t_j^v returns the smallest index if there are ties.

To minimize the least square error cost function, CART splits more where the function structure changes. As a result, more knots are identified in highly structured regions, and CART does not split where there are no significant changes in the response. In the beginning, the density is equal in each of the terminal nodes since we start with a limited designed set of initial points that are uniformly scattered in the domain. However, as we add more points in the promising regions, the density changes across terminal nodes.

Algorithm 3, presents our TK-MARS framework.

Algorithm 3 TK-MARS

1: Construct CART on data set;
$I = \{(x_1^i, \ldots, x_d^i, y_i) | \forall i = 1, \cdots, N\}$
2: Find the centroids for each $v \in V$ the set of terminal nodes
$c_j^v = \frac{1}{K_v} \sum_{k=1}^{K_v} x_j^k \quad \forall v = 1, \cdots, V, j = 1, \cdots, d$
3: Determine the index of the closest point to centroids in each dimension;
$t_j^v \in \text{argmin}_{k=1,\cdots,K_v} |x_j^k - c_j^v| \quad \forall v = 1, \cdots, V, j = 1, \cdots, d$
(in the case of having ties randomly select one; smallest index)
4: Fit MARS using eligible knot locations $x_j^{t_j^v}, \forall v = 1, \ldots, V, j = 1, \ldots, d.$

The proposed eligible knot location approach is customized to locate knots for the optimization purpose by capturing peaks and valleys beside obtaining optimized MARS with fewer basis functions. Locating knots, where the response changes more frequently or faster, identifies an optimal solution faster without focusing on the overall accuracy of the function estimation. TK-MARS does not necessarily focus on function behavior but an optimum point. In addition, MARS is sensitive to the order of the data points. Consequently, the regression tree helps to bring in the near optimal data points earlier, and that accelerates the performance of MARS.

3.2 Unimportant Variables

In this research, we introduce a new paradigm in black-box optimization by considering the significance of the input parameters. In real-world simulations, a large percentage of the input variables are unimportant. Pilla et. al. [12] shows that only 42 out of 1264 variables in a fleet assignment case study remain significant. In black-box simulators, there is a large set of input variables that may not be important to the output. Since we have no prior knowledge about the computer experiment and the underlying model, it is unknown whether a variable is important or not *a priori*.

In this research, we demonstrate the ability of the proposed method in identifying important variables. TK-MARS is a MARS-based non-interpolating surrogate model that constructs a parsimonious model using the forward selection and the

backward elimination procedures of MARS.

As we discussed in Section 2.3.1.2, the forward selection and the backward elimination procedures of MARS tend to choose a small subset of appropriate basis functions, which are defined by the eligible knot locations in different dimensions and minimize the lack of fitness. Hence, MARS is able to identify significant variables regarding the basis function selection procedures. This is practical for black-box functions in which not all input variables are important and are unknown *a priori*.

In Chapter 4, we conduct a set of experiments in order to show that the proposed approach has the ability to weed out unimportant variables of the black-box function and drop them automatically. We compare the results with RBF, the most common interpolation surrogate model in the literature, which considers all of the variables. Upon conducting the test, we ignore a fraction of variables in function evaluation and consider the remaining as important variables for computer experiments. In this way, the actual objective value is based on a fraction of important variables, not all of them. We expect TK-MARS to determine the important variables and to drop the unimportant ones.

Identifying unimportant variables is vital to black-box optimization, since, despite a large number of input parameters, only a small subset of those might affect the black-box funciton. Constructing an approach that can identify the significant variables, reduces the complexity of the surrogate model and, as a result, improves the optimization process.

3.3 Centroid-Based Dynamic Pool Generation

Candidate selection is an important step in the surrogate optimization Algorithm 1, as explained in Chapter 2. In this study, we apply an exploration-exploitation method to find the new candidate points, in order to update the data set until a predetermined maximum number of function evaluations is exhausted.

There are several candidate selection algorithms for surrogate optimization. EEPA [14] has the advantage of simultaneously finding the non-dominated candidates from the constructed pool and trading off the exploration of the uncovered regions and the exploitation of the regions of interest. We employ EEPA [14] for the candidate selection procedure for the following reasons:

- **Adding multiple points at a time:** EEPA selects multiple samples at a time to be added to the initial data set.
- **Representing extreme points that are multi-objective:** EEPA represents all the non-dominated points considering the exploration and exploitation as the two objectives of the Pareto frontier.
- **Identifying the uninteresting regions:** EEPA ignores the set of dominated points by pruning the regions that are worse both in exploration and exploitation from selected candidate points (Pareto frontier).

EEPA is explained precisely in Section 2.4.1.

Since the variables domain $D = \{x \in \mathbb{R}^d : a \leq x_j \leq b, \forall j = 1, \cdots, d\}$ is a continuous space, finding candidate points first requires a discretization approach. Therefore, a fixed pool construction is necessary to generate feasible data

points for the future candidate selection procedure. In this study, we propose a novel dynamic pool construction approach and some modifications to the distance measure in order to improve the overall performance of EEPA.

The standard EEPA algorithm applies a fixed random pool of points in the domain of the input space. However, generating a fixed set of random points cannot cover the space perfectly. As a result, it needs a large pool, which makes the surrogate optimization procedure inefficient as more exploration is needed . Further, in the presence of noise, even a large pool may not be sufficient. As a result, a dynamic pool construction that synthesizes the sample points as needed can facilitate the candidate selection procedure.

We assume a global optimum is in the feasible domain D.

On the other hand, the optimum of the black-box function is unknown. Given a fixed number of function evaluations, BSMS may not be a global optimum. Using the information about the function structure in each iteration helps with the gradual evolution of the pool, such that the pool converges to a global optimal solution. The convergence and the quality of the BSMS depend on the uncertainty level of the system and the available budget.

Consider a set of N uniformly distributed points, R, spread in the parameters' feasible domain. The selected new candidate points P are added to the pool R, repeatedly. We propose a novel approach in order to achieve a good representation of the solution space and converge to a global minimum, effectively. Regarding the first step of TK-MARS, fitting CART on the already evaluated points, we have the set of terminal nodes V (representing the partitions).

These terminal nodes represent different regions of the solution space, where the underlying function has different structures. Consider the centroid c_v of each partition $v \in V$. The quality of these centroids from the TK-MARS knot selection procedure is discussed in Section 3.1, as a result of the regression tree logic. Figure 3.2 shows the TK-MARS centroids that we use for dynamic pool generation.

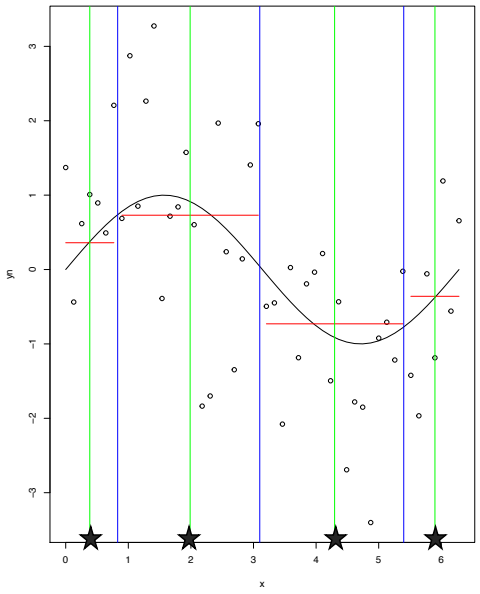


Figure 3.2: TK-MARS Centroids for Dynamic Pool Generation

Since the centroids of the terminal nodes are close to peaks and valleys of the underlying black-box function, they can be good candidates to be added to the fixed random pool generated, repeatedly. At the end of each iteration, the solution space is defined as $R = R \cup C$ where R is the fixed random pool, and C is the set of centroids of the terminal nodes – $|C| = |V|$.

The pre-initialized fixed random pool R and the newly added points C result in an exploration-exploitation balance for the solution space discovery, as well as convergence to a global optimum. While the initial random points in the pool serve as the exploration, dynamically added points around the interesting regions is exploitation. As more information is obtained from the structure of the function and the feasible space, this strategy increases the chance of generating candidate points in different partitions of the solution space. Here the exploration finds points from a discretized continuous space. Any points can be selected for the exploration part. However, having a large pool decreases the probability of choosing the centroids (the set of promising points) as the next candidates. Considering a reasonable initial pool size allows the approach to add effective points later. A larger initial pool assigns more weight on exploration.

Over time, the centroids effectively move in the solution space. We add more points near BSMS using the tree logic, instead of using a large fixed random pool representing the whole feasible region. As a result, the candidate points are effectively near potential optima.

There are candidate selection techniques in the literature as we discussed in Section 2.4 that generally either solve an optimization subproblem or generate data points based on random perturbations around a potential optimum. Those, however, might not work well for noisy data sets. Our dynamic pool generation strategy is not random as it follows the pattern of the function fluctuations more intelligent than random perturbation. Note that the centroids are not necessarily in the pool, even though they could be close to the promising points. As a result,

some of the centroids lie around the BSMS. Consequently, adding centroids may be similar to perturbation around BSMS [41]; however, it is not probabilistic.

3.4 Cosine Similarity Exploration Metric

In Section 2.4.1, we describe the candidate selection algorithm we apply in this study. EEPA is an exploration-exploitation Pareto approach for finding the best candidate points. To identify the non-dominated set of solutions, EEPA defines a distance metric for exploration and uses estimated objective value for exploitation. Different distance metrics can be applied in EEPA, such as Euclidean and Cosine similarity. Cosine similarity calculates the Cosine of the angle between two vectors by using inner product space $\cos(X, \hat{X}) = 1 - \frac{X}{\|X\|}.\frac{\hat{X}}{\|\hat{X}\|}$. Cosine provides information about the alternative directions. To find the perfect right angle of a direction, the Cosine measure needs to be zero.

A perpendicular direction of the points to the already evaluated points gives more information on the unexplored regions. The key challenge is that the Cosine similarity metric cannot be calculated for points but vectors. We have a set of already evaluated points I and a random pool R from which the next candidate points are selected. To calculate the Cosine distance of the potential candidates from the already evaluated points, we need to consider the points as origin-starting vectors, and then calculate the angle between the vectors. Hypothetically, the origin can be anywhere in the space. However, to find the right angle, we consider the center of the feasible domain as the origin.

As a result, a standardization procedure is required to calculate the Cosine metric, as given by Algorithm 4.

Algorithm 4 Cosine Pseudocode

1: $X = \{a \leq x_j^i \leq b \mid j = 1, \cdots, d\,, i = 1, \cdots, N\}$
2: $\hat{X} = \{a \leq \hat{x}_j^i \leq b \mid j = 1, \cdots, d\,, i = 1, \cdots, |R|\}$
3: $O_j = \frac{b-a}{2}\ \forall j = 1, \cdots, d$
4: $x_j^i = x_j^i - o_j : \forall j = 1, \cdots, d\,, i = 1, \cdots, N$
5: $\hat{x}_j^i = \hat{x}_j^i - o_j : \forall j = 1, \cdots, d\,, i = 1, \cdots, |R|$
6: $\cos(X, \hat{X}) = 1 - \frac{X}{\|X\|} \cdot \frac{\hat{X}}{\|\hat{X}\|}$
7: **return** $\cos(X, \hat{X})$

Algorithm 4, calculates the Cosine angle between the vectors originated from the center of the space.

In this research, we use the standardized Cosine similarity as the exploration metric in EEPA. We believe that the standardized Cosine is better at discovering unexplored regions than the regular Euclidean distance. The standardized Cosine discovers the candidates that are uniformly scattered in the domain. Considering the center of the space as the origin helps the exploration, as it maintains its knowledge about the different part of the space as in Figure 3.3.

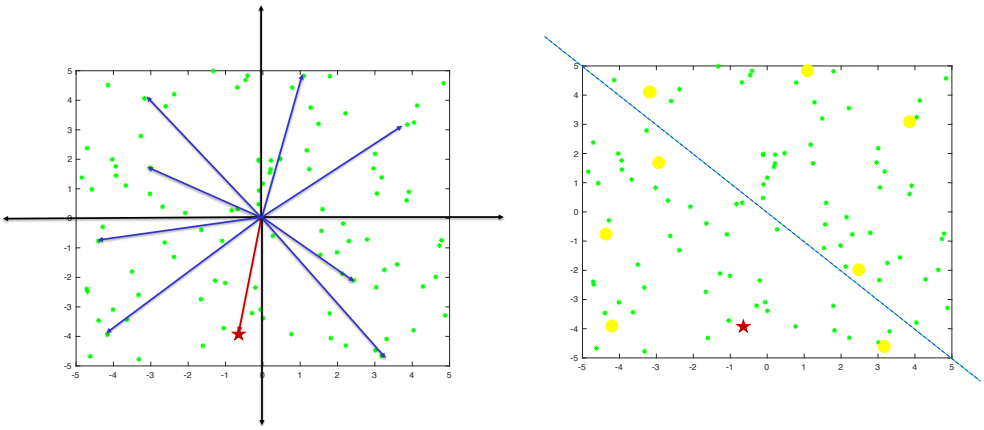


Figure 3.3: Cosine Similarity with Center of the Space as the Origin

Considering 0 as the origin when the domain is not symmetric, the selected candidate points are not chosen equally around the domain as in Figure 3.4.

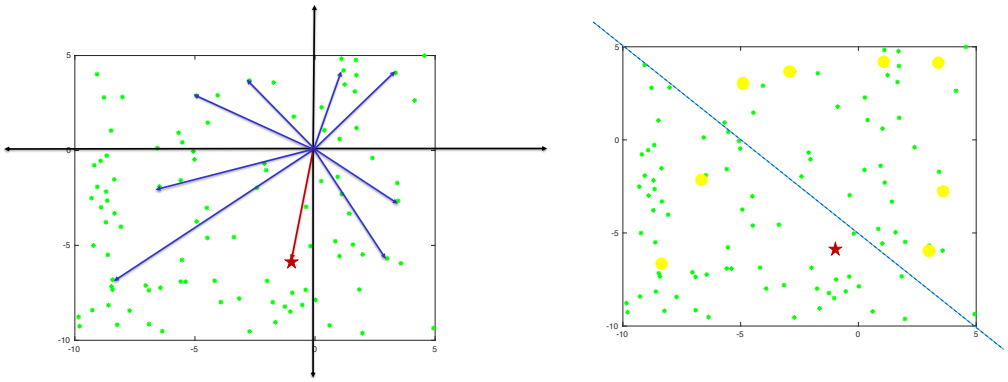


Figure 3.4: Cosine Similarity with 0 as the Origin

3.5 Deterministic Black-Box Systems

In this section, we present our surrogate optimization algorithm for expensive black-box functions, Algorithm 5, following the proposed contributions we described previously in this chapter. Aggregating the concepts in this dissertation, Algorithm 5 demonstrates our proposed surrogate optimization for deterministic black-box functions.

We assume the black-box function is noise-free, i.e., if we execute a single input several times, we will get the same output.

Algorithm 5 Deterministic Approach

1: Sample initial design space $I = \{x_j^i \in D \mid \forall j = 1, \cdots, d, i = 1, \cdots, N\}$
2: Randomly generate m uniform random points in the box region;
   $R = \{u_j^i \mid a \leq u_j^i \leq b,\ \forall\ j = 1, \cdots, d, i = 1, \cdots, m\}$
3: Evaluate initial data set I, $f(x^i), \forall x^i \in I$
4: **while** Termination criteria is not satisfied **do**
5: Construct CART on initial data set I
6: Find the centroids for terminal nodes
   $c_j^v = \frac{1}{K_v} \sum_{k=1}^{K_v} x_j^k\ , \forall v = 1, \cdots, V\ , j = 1, \cdots, d$
7: Construct a surrogate model on I (TK-MARS/RBF)
8: Add centroids to the R; $R = R\ \cup\ C$ where $C = \{c^v \mid \forall v \in V\}$
9: Optimizer on R to determine new candidate set of points P
10: Evaluate new selected candidate points $\forall x \in P$ to find $f(x)$
11: Update initial data set $I = I \cup P$
12: Find BSMS
13: **end while**

In step 1, N points are designed in the feasible solution space to spread throughout the domain $[a, b]$. Next, the objective value is evaluated for the N initial data points. N equals to $d + 1$ at the beginning. In this research, we employ Latin Hypercube Design for the initialization of the input space, which is considered

as a low-discrepancy sequence but not typically orthogonal, and Sobol, which is a quasi-random low-discrepancy sequence with uniformity properties. The type of design of experiment method we use for initialization is one of the algorithm parameters, which has two levels of OA and Sobol design. In step 5, the algorithm applies CART partitioning strategy to partition the solution space based on the function structure in Section 3.1. Following the partitioning step, a surrogate model has to be fitted. In this study, we focus on MARS and present a modified version of it, TK-MARS, by developing a new knot-selection approach employing the centroids of the partitions obtained in step 6. In step 2, to discretize the solution space, the algorithm generates a fixed number of random points first and in step 8 dynamically improves the representation of the feasible domain by adding the centroids from step 6. This novel approach provides more information around the promising neighborhood of potential optima. In step 9, we apply EEPA on the generated pool to select the best subset of candidate points in order to add to the set I. We repeat all the steps until the maximum number of function evaluations is exhausted.

3.6 Uncertainty in Black-Box Systems

In Section 3.5, we assume that the system is noise-free. However, this assumption is not valid in many real-world applications. In some expensive computer simulations, there is inherent noise associated with the system. Most of the existing methods cannot handle uncertainty [21, 52, 53].

We relax the noise-free assumption in this section, hence, the response obtained from simulation contains uncertainty. $\tilde{f}(x) = f(x) + \epsilon$ where $\tilde{f}(x)$ is the output of the black-box system that contains uncertainty.

This means the output of the simulation is a stochastic function, which differs from the true function value. Every time a point is evaluated it comes out with a new response. Consequently, we have different outputs for a single simulation. The goal still is to minimize the true objective function as in Equation 3.4; however, only the simulated noisy function values (actual objective value) are available.

$$\min f(x) \tag{3.4}$$

$$s.t.$$

$$a \leq x \leq b, \forall x \in \mathbb{R}^d. \tag{3.5}$$

$f(x)$ is the true objective value. We assume that the feasible region, D, $D = \{x \in \mathbb{R}^d : a \leq x_j \leq b, j = 1, \cdots, d\}$, is continuous. In addition, we assume that the uncertainty or noise are independent identically distributed with mean 0 and variance of σ^2, $\tilde{f}(x) \sim (f(x), \sigma^2)$, following an unknown distribution. Stochastic black-box systems are rarely addressed in surrogate optimization literature as we discussed in Section 2.6.

As we discussed earlier in Section 2.3, different types of surrogate models can be chosen for surrogate optimization. Interpolating methods need more points in

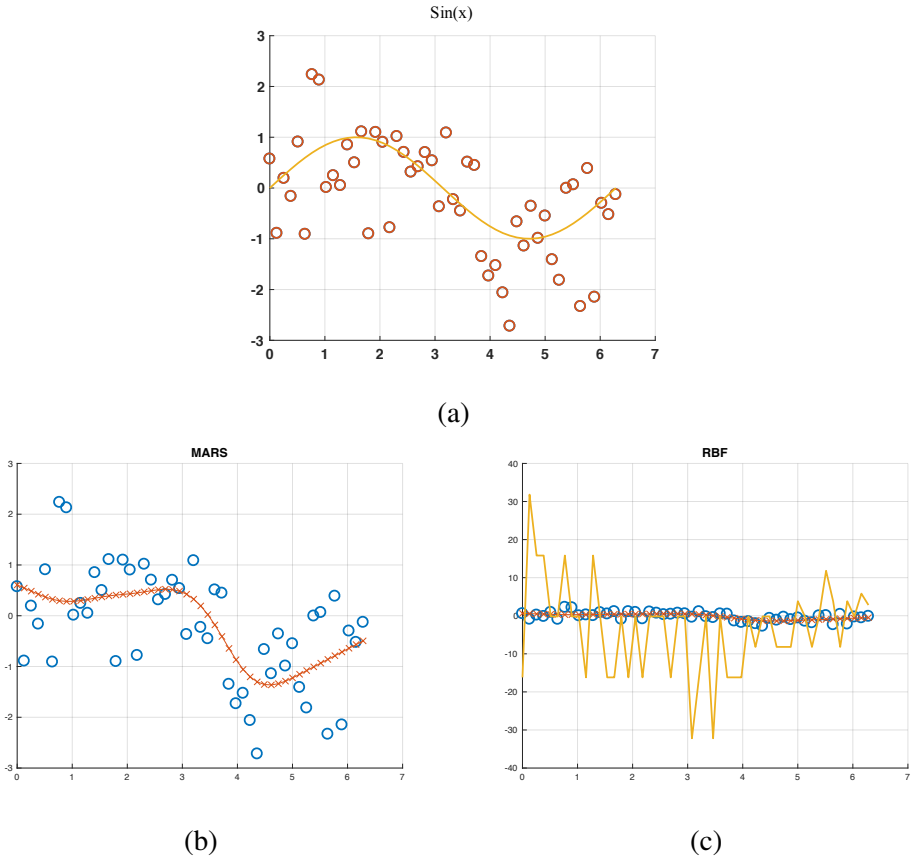
order to estimate a more accurate model and are commonly used for the deterministic situations. Since interpolation-based models pass through all the points and need the exact true value for each point, they may result in a highly oscillating surrogate model for noisy systems. However, non-interpolating methods pass among all the points and do not necessarily need exact values of the points. They minimize the average error of the estimated values. These models are good options, especially when the objective function contains noise.

Jones [45] shows that non-interpolating surfaces, such as quadratic surfaces, can be unreliable as they do not capture the shape of the function. However, he does not study the stochastic situation and focuses only on deterministic black-box functions.

In highly noisy systems, a non-interpolating approach is usually preferred to avoid oscillations of the fitted surface, when an interpolating approach is forced to go through all data points. Differences between interpolating and non-interpolating models are investigated in this research, too.

In Figure 3.5, we show how an interpolating model, as in Figure 3.5c, and a non-interpolating model, as in Figure 3.5b, perform for a simple $\sin$ function with noise and recognize the true underlying behavior under uncertainty.

As we mention earlier, we develop a new version of Multivariate Adaptive Regression Splines, TK-MARS, for the purpose of surrogate optimization of expensive black-box functions. However, in order to analyze the performance of the proposed method, different variants of RBF modeling, as an interpolating surrogate, are employed in the implemented surrogate optimization algorithm. For de-



(a)

(b)

(c)

Figure 3.5: Interpolating versus non-Interpolating models for noisy sin function

terministic black-box systems, our proposed TK-MARS is competitive with RBF, as the fraction of significant variables decreases.

We employ TK-MARS in a surrogate optimization algorithm with a Pareto-based optimizer as in Algorithm 5. In this section, we study different scenarios for black-box systems with uncertainties. The proposed algorithm implements a new way of handling noise in the objective function. If no noise is present in the objective function, an interpolation is a good choice as a surrogate model. In order

to capture the single simulation with two or more different outputs, the interpolation method needs to pass through all the points with a large slope. Consequently, interpolation-based surrogate models may result in highly fluctuated approximations for data points include uncertainty. The TK-MARS approximation becomes less oscillating than the interpolation-based RBF. It resembles the true function behavior better than the interpolation. Furthermore, because of the less oscillating behavior of TK-MARS, tracking the minimum of the approximation is often easier for TK-MARS than for an interpolation-based surrogate model.

Consequently, we expect TK-MARS to be more capable of handling the stochastic black-box functions as a non-interpolating method, which mainly focuses on optimizing the promising regions. However, we propose three different approaches to reduce the uncertainty associated with the black-box functions, by either using a non-interpolation or by an interpolation-based surrogate model. The first approach is to employ the deterministic surrogate optimization approach Algorithm 5 introduced in Section 3.5. An alternative approach to cancel out the uncertainty effect is replicating the function evaluation for each point.

Evaluating a point once with a stochastic black-box function may not be reliable and accurate. Hence, doing more than one evaluation for the same point gives more accurate information about the true objective value. We propose two different replication strategies as the alternative approaches for stochastic black-box optimization, in order to minimize the output uncertainty. Based on the level of uncertainty and the maximum number of function evaluations, one may move along between exploring more points with single evaluation and exploiting a few

points by replicating more than once.

Although with replication, the effect of noise decreases and the sampled mean converges to the true function value, it requires more function evaluations when the uncertainty level is high. As a result, the replication effect is not significant; exploration, on the other hand, is more effective.

We assume that the actual objective function of the black-box is $\tilde{f}(x)$ where $\tilde{f}(x) \sim N(f(x), \sigma^2)$. Suppose the number of replication is r. Let $\bar{f}(x)$ be the sample mean of $\tilde{f}(x)$ after r replications. That is, $\tilde{f}(x^i) = f(x^i) + \epsilon, \ \forall i = 1, \cdots, |I|$.

$$\bar{f}(x^i) = f(x^i) + \frac{\sum\limits_{\substack{x^j = x^i \\ j \leq i}} \epsilon(x^j)}{\sum\limits_{\substack{x^j = x^i \\ j \leq i}} (x^j = x^i)}, \ \forall \, i = 1, \cdots, |I| \tag{3.6}$$

$\bar{f}(x)$ in Equation 3.6 follows $N(f(x), \frac{\sigma^2}{\sqrt{r}})$ based on the central limit theorem. $\epsilon(x^j)$ is the uncertainty component of $\bar{f}(x^j)$. The second component in Equation 3.6 corresponds to the mean of replications' uncertainties for a sample point. $(x^j = x^i)$ is an indicator function that counts the number of replications for a single point. Hence, when the uncertainty level is high, the replication does not reduce the variance significantly. As a result, exploring more single evaluated points will be a better option.

Next, in Sections 3.6.1, 3.6.2, and 3.6.3, we explain three approaches for black-box optimization in noisy applications.

3.6.1 Deterministic Approach: No-Replication

In this section, relaxing the no-noise assumption, we apply the proposed deterministic approach, Algorithm 5, and adapt it for stochastic systems. In this approach, we use one evaluation for each sample point. The No-Replication Algorithm 5 exhausts the function evaluation limit for exploring different points and having a single evaluation for each of them.

Since the data points include uncertainties associated with the black-box function, a single evaluation may not be reliable and does not show the true output value of the black-box function. Consequently, the deterministic approach may not be adequate to handle the uncertainty effect and, as a result, mislead the optimization process.

Adding noise may yield a false function value; hence, the BSMS is different from a true optimal. The challenge is the model might get trapped in the artificial optima.

This approach is sufficient for the systems with a low level of uncertainty and cannot handle a higher level of uncertainties. The deterministic approach may not be robust to the uncertainties associated with the black-box system.

3.6.2 Fixed-Replication Approach

The No-Replication approach uses a single evaluation of each point. The other extreme is to evaluate the selected points multiple times. In this scenario, each sample point is evaluated a fixed number of times r, where $r > 1$.

Replication mitigates the noise effect. Based on the central limit theorem, the average of r random values ϵ, distributed with mean 0 and standard deviation of σ, follows a normal distribution of $N(0, \frac{\sigma^2}{\sqrt{r}})$. That is, the variance of the average is smaller than that of a single sampled random variable. Consequently, replication and averaging over the replicated function values is a good estimation of the mean of a noisy function.

Let us define r as $r = \sum_{\substack{x^j = x^i \\ j \leq i}} (x^j = x^i),\ \forall i = 1, \cdots, |I|$. In other words, r is the number of equal pair of sample points from the set of already evaluated points, I.

Algorithm 6 represents the Fixed-Replication approach.

Algorithm 6 Fixed-Replication Approach

1: Sample initial design space $I = \{x_j^i \in D \mid \forall j = 1, \cdots, d, i = 1, \cdots, N\}$
2: Randomly generate m uniform random points in the box region;
$R = \{u_j^i \mid a \leq u_j^i \leq b,\ \forall\ j = 1, \cdots, d, i = 1, \cdots, M\}$
3: **while** Termination criteria is not satisfied **do**
4: Evaluate initial data points each r times to find $\bar{f}(x), \forall x\ \in\ I$
5: Construct CART on initial data set I
6: Find the centroids for terminal nodes
$c_j^v = \frac{1}{K_v}\ \sum_{k=1}^{K_v} x_j^k,\ \forall v = 1, \cdots, V, j = 1, \cdots, d$
7: Construct a surrogate model on I (TK-MARS/RBF)
8: Add centroids to R; $R = R\ \cup\ C$ where $C = \{c^v \mid \forall v \in V\}$
9: Optimizer (EEPA) on R to determine new candidate set of points P
10: Evaluate new selected candidate points each r times to find $f_1(x), \cdots, f_r(x), \forall x\ \in\ P$
11: Update initial data set $I = I \cup P$
12: Find BSMS
13: $t := t + 1$
14: **end while**

In steps 4 and 10, either each replicated point can be considered as a new sam-

ple point, or the average of r replications can be taken as a single point. However, interpolation-based surrogate models cannot handle a point with multiple different objective values due to the singularity issue. On the other hand, non-interpolating models, such as MARS, can consider the same point with different response values. As a regression-based model, MARS does not necessarily pass through all the points but rather tends to minimize the error. In the experiments in Chapter 4, we consider the average value over replications for interpolating models. For non-interpolating models, we study the average of replications (TK-MARS-Avrg) and also consider all the replications as new sample points (TK-MARS-Keepall).

Replicating all candidate points a fixed number of times is not efficient and wastes function evaluations. That is because all of the already evaluated points, no matter if promising or not, are replicated r times. The Fixed-Replication approach is sufficient for systems with a higher level of uncertainty and wastes unnecessary function evaluations when the uncertainty level is low. In the Fixed-Replication approach, the algorithm exhausts the function evaluations faster having multiple evaluations at a time. Consequently, it leads to less exploration and more reliability.

In the presence of noise, the Fixed-Replication approach is more robust in comparison with the deterministic approach. However, the number of replications, r, has to be predetermined. It is always hard to choose the optimum number of replications, as it depends on the noise level and the underlying function characteristics.

3.6.3 Smart-Replication Approach

So far, we explained No-Replication and Fixed-Replication approaches. A smarter strategy, in between, is to replicate not all of the candidate points but only the promising points for optimization. Following this strategy, in this section, we propose an approach for replication, using a hypothesis testing based on confidence intervals. In this approach, the promising points, close to the best sampled mean solution, are replicated. The number of replications is determined by the algorithm itself, following the hypothesis rule. Smart-Replication automatically decides the number of replications for each selected candidate point.

Assume that the current BSMS is x^o. For each selected candidate point x_i, Smart-Replication considers the following null hypothesis and stops replicating x_i, if it can reject the null hypothesis.

> **$\mathbf{H}_o$**: if $\bar{f}(x^o) < \bar{f}(x^i)$, where $x^i \neq x^o$, $\forall i = 1, \cdots, |I|$, then $f(x^o) > f(x^i)$.
>
> **$\mathbf{H}_1$**: if $\bar{f}(x^o) < \bar{f}(x^i)$, where $x^i \neq x^o$, $\forall i = 1, \cdots, |I|$, then $f(x^o) < f(x^i)$.
>
> **Decision rule**: if $CI_{low}(f(x^i)) \geq CI_{up}(f(x^o)), \forall x^i \in I$, Reject H_o.
>
> **Conclusion**: Even though the objective value of BSMS, $\bar{f}(x^o)$, is smaller than the objective value of x^i, $\bar{f}(x^i)$, the true objective value of x^i, $f(x^i)$, is smaller than the true objective value of x^o, $f(x^o)$. Hence, we are $(1 - \alpha)\%$ confident that more replications are required for x^i.

For each point that is evaluated r^i times, $r^i \geq 2$, we calculate the standard

deviation and the confidence interval with a significance level of α.

$$std(x^i) = \sqrt{\frac{1}{r^i - 1} \sum_{k=1}^{r^i} (\tilde{f}(x^k) - \mu)^2}$$

where $\mu = \frac{1}{r^i} \sum_{k=1}^{r^i} \tilde{f}(x^k)$.

Now, we calculate the confidence interval, $CI(x^i) = (\mu \mp t_{\frac{\alpha}{2}} \frac{std(x^i)}{\sqrt{r^i}}), \forall i = 1, \cdots, |I|$. To select the promising points, we prune the candidate points based on the lower bound of the confidence interval CI_{low}. If CI_{low} of a point is less than the threshold, i.e. $threshold = CI_{up}(BSMS)$, it is selected as a promising point to be replicated. The promising points are the candidate points that are close to the current BSMS, i.e., below the threshold, Figure 3.6. Since the number of function evaluations is limited, and the experiments are costly, we consider a maximum number of replications for the promising points r_{max}.

The smart approach behaves similar to the deterministic, No-Replication, approach when the uncertainty level is low and similar to the Fixed-Replication approach when the uncertainty level is high. For a higher noise level, the variance is larger. It makes the condition of CI ineffective, and, as a result, more replications are required. However, it can replicate points up to the maximum number of replications r_{max}, which needs to be large and determined in advance.

Consequently, Smart-Replication is efficient in both ways, i.e. low/high noise. That is because it recognizes if the system is deterministic or stochastic, and decides about replications automatically.

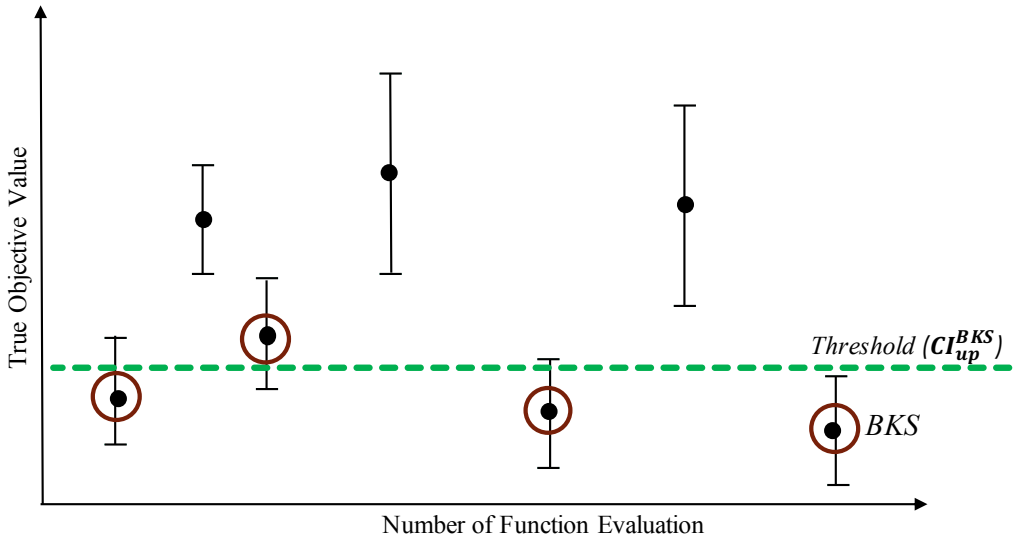


Figure 3.6: Smart-Replication Approach Illustration

3.7 Measure of Performance

In this section, we propose two measures of performance for deterministic and stochastic systems. since the function evaluations are expensive, the number of function evaluations before finding an optimal solution is a good metric for measuring the performance of an algorithm. However, obtaining a global optimum cannot be guaranteed. A metric that can quantify the convergence pattern of an algorithm is more plausible in the context of black-box optimization.

3.7.1 Area Under the Curve: AUC

First, we propose the area under the curve (AUC) metric in Definition 1.

Definition 1 (Area Under the Curve – AUC)**.** *Given the best sampled mean solu-*

Algorithm 7 Smart-Replication Approach

Sample initial design space $I = \{x_j^i \in D \mid \forall i = 1, \cdots, d, j = 1, \cdots, N\}$
Randomly generate m uniform random points in the box region;
$R = \{u_j^i \mid a \leq u_j^i \leq b,\ \forall\ j = 1, \cdots, d, i = 1, \cdots, M\}$
while Termination criteria is not satisfied **do**
  $\mu^i = \frac{1}{r^i} \sum_{k=1}^{r^i} x^k$
  $std(x^i) = \sqrt{\frac{1}{r^i - 1} \sum_{k=1}^{r^i} (x^k - \mu^i)^2}$
  $CI(x^i) = (\mu \mp t_{\frac{\alpha}{2}} \frac{std(x^i)}{\sqrt{r^i}}), \forall i = 1, \cdots, M$
  while $CI_{low}(f(x^i)) \geq CI_{up}(f(x^o))$ & $r^i \leq r_{max}$ **do**
    Evaluate x^i, $r^i = r^i + 1$
    Update $std(x^i), \mu^i, CI(x^i)$,
  end while
  Construct CART on initial data set I
  Find the centroids for terminal nodes
  $c_j^v = \frac{1}{K_v} \sum_{k=1}^{K_v} x_j^k,\ \forall v = 1, \cdots, V, j = 1, \cdots, d$
  Construct a surrogate model on I (TK-MARS/RBF)
  Add centroids to the R; $R = R \cup C$ where $C = \{c^v \mid \forall v \in V\}$
  Optimizer (EEPA) on R to determine new candidate set of points P
  Update initial data set $I = I \cup P$
  Find BSMS, $(x^o, f(x^o))$
  t:=t+1
end while

tion (BSMS) found by an algorithm after each function evaluation. Let $f(x^oi)$ be the true objective value of BSMS, where x^oi is the BSMS after i function evaluations. Let $f^{min} = \min_{i=1,\ldots|I|}(f(x^{oi}))$ and $f^{max} = \max_{i=1,\ldots|I|}(f(x^{oi}))$. Using the normalized objective value of BSMS:

$$f(x^{oi}) = \frac{f(x^{oi}) - f^{min}}{f^{max} - f^{min}}\ \forall i = 1, \cdots, |I|,$$

the AUC is

$$AUC = \sum_{i=1,\cdots,|I|} f(x^{oi}) \tag{3.7}$$

AUC comprises both the quality of BSMS, as well as the time that the algorithm finds it.

AUC works well for the deterministic cases, as the objective value of the BSMS monotonically decreases over time. However, this may not hold for stochastic black-box systems, as the curve consists of jumps. In these cases, the stability of BSMS represents the robustness. Even though the jumps in early iterations are tolerable, we expect a stable behavior towards the end of the evaluation. Therefore, next, we propose a metric for stochastic black-box systems that is able to monitor the stability and jump locations across the number of function evaluations. A good algorithm is the one with fewer and shorter jumps, in which after a reasonable number of function evaluations, the results are reliable.

3.7.2 Maximal True Function Area Under the Curve: MTFAUC

In this section, we propose a new performance measure for stochastic black-box optimization. To consider the instability in the metric, we consider the maximum objective value of BSMS obtained among all BSMS found forward. Subsequently, we define the maximal true function area under the curve (MTFAUC) as following:

Definition 2 (Maximal True Function Area Under the Curve – MTFAUC)**.** *Given*

the best sampled mean solution (BSMS) found by an algorithm after each function evaluation. Let $f(x^{oi})$ be the true objective value of BSMS, where x^{oi} is the BSMS after i function evaluations. Let $a = \min_{i=1,\dots|I|}(f(x^{oi}))$ and $b = \max_{i=1,\dots|I|}(f(x^{oi}))$. Using the normalized objective value of BSMS

$$f(x^{oi}) = \frac{f(x^{oi}) - a}{b - a} \; \forall i = 1, \cdots, |I|,$$

$\bar{f}(x)$ (Equation 3.6) is the sample mean of the observed objective value after r replications. Let $x^{oi} = \text{argmin}_{j=1,\cdots,i} \, \bar{f}(x^j)$ be the BSMS after i number of function evaluations and $\bar{f}(x^{oi})$ be its estimated objective value. Consider $\hat{j}(i)$ as the maximum objective value of BSMS obtained among all BSMS found forward:

$$\hat{j}(i) \in \underset{j=i,\cdots,|I|}{\text{argmax}} \, f(x^{oj}) \tag{3.8}$$

The MTFAUC is

$$MTFAUC = \sum_{i=1}^{|I|} \frac{f(x^{\hat{j}(i)}) + f(x^{\hat{j}(i-1)})}{2} \tag{3.9}$$

MTFAUC comprises the instability, by using $\hat{j}(i)$ to penalize the jumps. Hence, MTFAUC deteriorates if the convergence pattern highly fluctuates. A more stable algorithm towards uncertainty has a lower MTFAUC value.

Chapter 4

Computational Experiments

4.1 Implementation

The proposed methods are implemented in R[1]. The original version of MARS is in C. The proposed knot preprocessing approach for MARS is part of an R code. We use RBF toolbox for MATLAB/OCTAVE [54].

4.2 BBOB Test Functions

We compare the performance of the proposed algorithms on four test problems, taken from a collection of well-known global optimization test functions [55, 56]. In this study, we first evaluate the deterministic black-box functions, and then, examine the performance of the proposed algorithms under uncertainty. Table 4.1

[1] https://www.r-project.org/

describes the characteristics of the selected test functions.

- **Rosenbrock:** non-convex uni-modal with interactions
- **Rastrigin:** non-convex multi-modal well-structured for the placement of the optima
- **Sphere:** unimodal convex
- **Levy:** non-convex multi-modal

Table 4.1: Test Functions Definition

Function	**Formulation**	**Range**	**Global Min**
Rosenbrock Fig. 4.1	$f(x) = \sum_{i=1}^{d-1}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	[-5,10]	$f(x*) = 0$ $x* = (1, \cdots, 1)$
Rastrigin Fig. 4.2	$f(x) = 10d + \sum_{i=1}^{d}[x_i^2 - 10cos(2\pi x_i)]$	[-5.12,5.12]	$f(x*) = 0$ $x* = (0, \cdots, 0)$
Sphere Fig. 4.3	$f(x) = \sum_{i=1}^{d} x_i^2$	[-5.12,5.12]	$f(x*) = 0$ $x* = (0, \cdots, 0)$
Levy Fig. 4.4	$f(x) = \sin^2(\pi w_1)$ $+ \sum_{i=1}^{d-1}(w_i - 1)[1 + 10\sin^2(\pi w_i = 1)]$ $+(w_d - 1)^2[1 + \sin^2(2\pi w_d)]$	[-10,10]	$f(x*) = 0$ $x* = (1, \cdots, 1)$

4.3 TK-MARS Study

We start by comparing the proposed TK-MARS surrogate model with the original MARS, which uses the evenly-spaced knot selection approach [9].

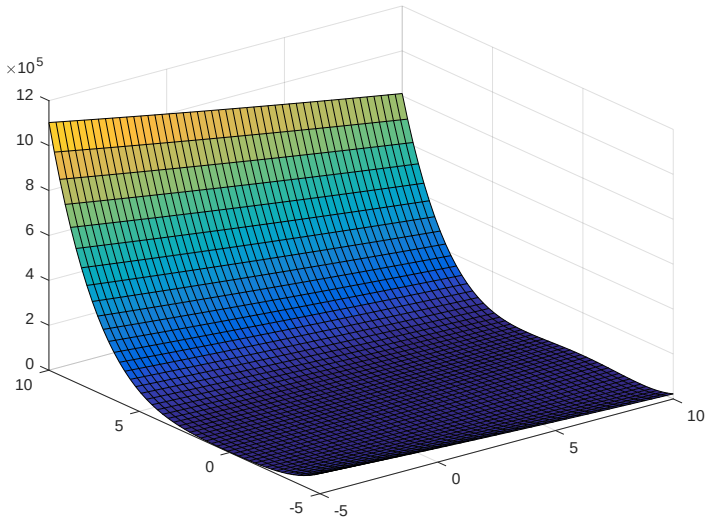


Figure 4.1: Rosenbrock Function

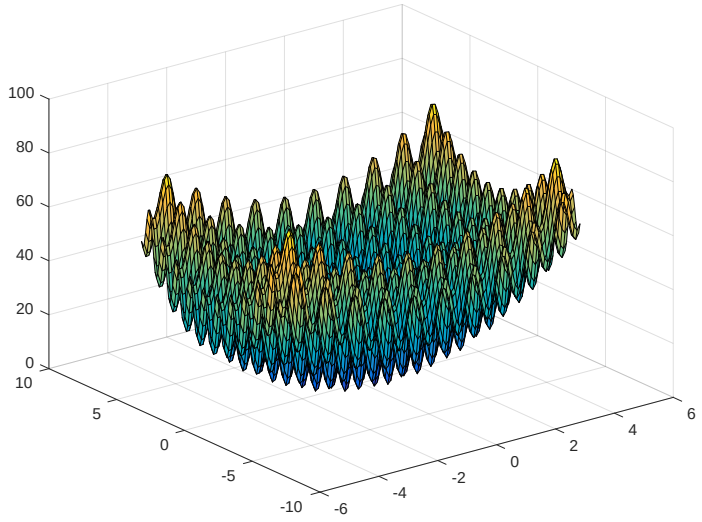


Figure 4.2: Rastrigin Function

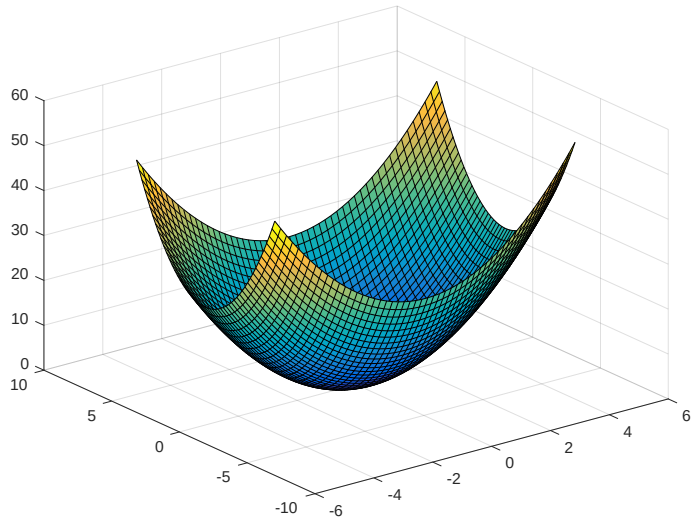


Figure 4.3: Sphere Function

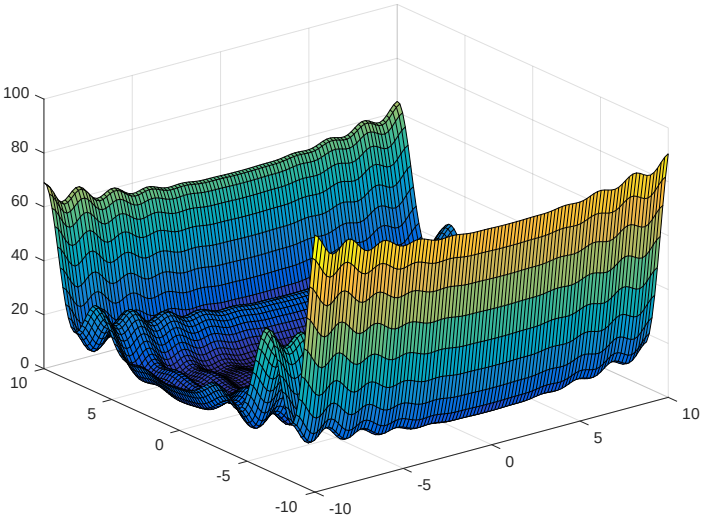


Figure 4.4: Levy Function

4.4 Parameters Setting

The input parameters of the MARS are presented in Table 4.2. We consider all different combinations of these parameters for the four test functions.

Table 4.2: MARS Parameters

Parameters
d = Number of X variables
T = Number of knots in each dimension
$\|I\|$ = Number of initial observations
$maxIA$ = maximum number of interaction terms per basis function
(1: no interaction, 2: two way interactions, 3: three way interactions)
M_{max} = maximum Number of MARS basis functions in approximation
$Alg3 = 0$ do not run Algorithm 3 (Backwards Deletion) of MARS
$Alg3 = 1$ do run Algorithm 3
$Robust = 0$ do not implement robust selection
$Robust = 1$ implement robust selection
TOL = tolerance (typically .1-.5)

The parameters of RBF are: (1) the type of the basis functions (Multiquadric (MQ), Gaussian (G), Cubic (C), Thin Plate Spline (TPS)), (2) the positive constant parameter ω value, and (3) a binary flag $poly$ to indicate whether or not the polynomial term needs to be added. Table 4.3 shows different values of the parameters to be studied in the experiments.

The initial set I of 35 points with $d = 27$ independent variables, is designed for a single black-box objective function, using a Latin Hypercube Design (LHD) low discrepancy method. The optimization problem is box-constrained; the model is presented in Equation 1.2. A uniform Random pool of 1000 points is designed for the candidate selection procedure, corresponding to the range of each test func-

Table 4.3: RBF parameters

RBF Type	Poly	ω
MQ	0	2
G	0	2
C	1	2
TPS	1	2

tion.

4.4.1 TK-MARS VS. Original MARS

First, we compare the original MARS with the proposed TK-MARS to show the advantage of the new knot preprocessing approach for surrogate optimization. The number of eligible knot locations in each dimension T is usually preset to a static value in MARS. The optimum value of T cannot be determined in advance, hence, we study the results for a small $T = 10$, a medium $T = 20$ and a large $T = 50$. To ensure a fair comparison between the original MARS and TK-MARS, we consider a dynamic number of eligible knot locations T for the original MARS. Specifically, after each iteration of the original MARS, we construct a tree using CART, as also done in TK-MARS. Then, we set T to be the number of terminal nodes, *leaves*, in this tree. Having both the original MARS and TK-MARS dynamically set T based upon the number of leaves from CART is more appropriate for comparing the two algorithms.

Finally, we perform TK-MARS on the test functions in order to show the improvement in identifying promising knot locations. The results are provided

in Figures 4.5-4.8. As one can see, the tree-based knots approach improves the optimization process, significantly.

To clarify, $T = 10$ is adequate for MARS in early iterations to capture the data structure. However as the number of sample points increases, $T = 10$ is insufficient and ultimately underfits the underlying function. On the other hand, with $T = 50$ MARS becomes more complex and overparameterized in early iterations. As a result, the optimization becomes worse and converges slower. In the case of T being set to the number of terminal nodes in CART, both TK-MARS and the original MARS have the same number of knots but in different locations. From the figures, TK-MARS lands the knots in the promising locations, using CART. As a result, TK-MARS properly captures the peaks and valleys, so the optimization converges faster.

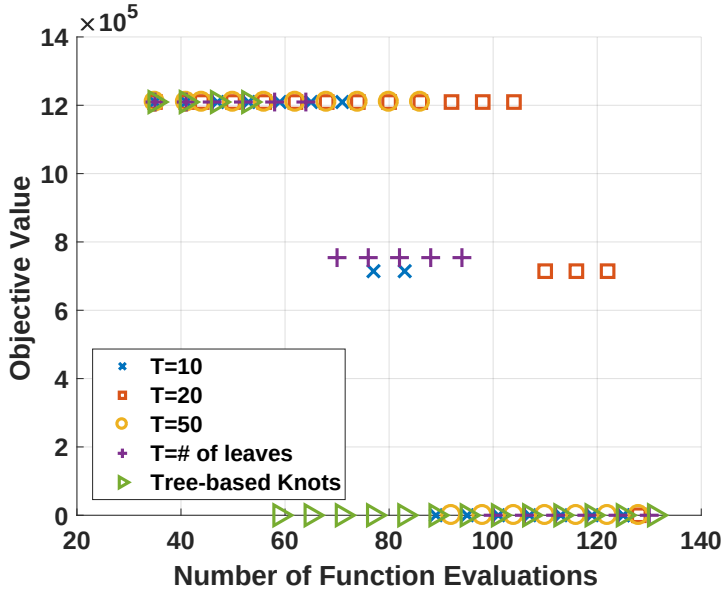


Figure 4.5: TK-MARS vs. Original MARS with different number of eligible knot locations – Rosenbrock function

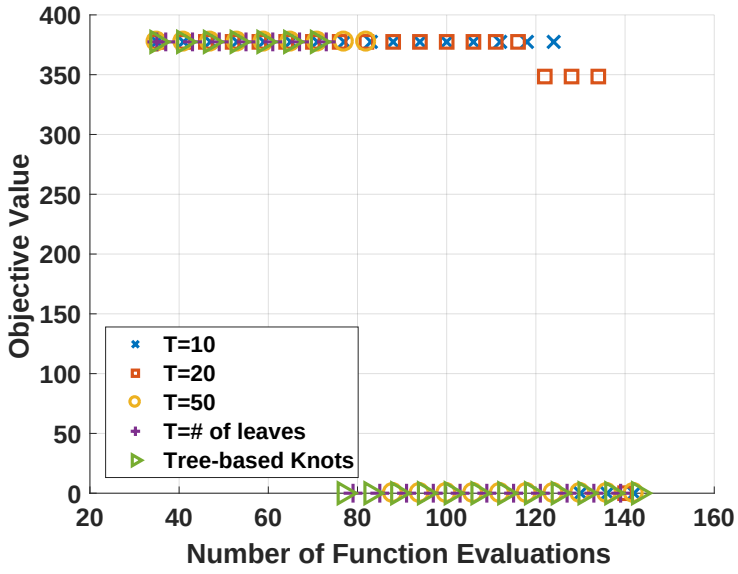


Figure 4.6: TK-MARS vs. Original MARS with different number of eligible knot locations – Rastrigin function

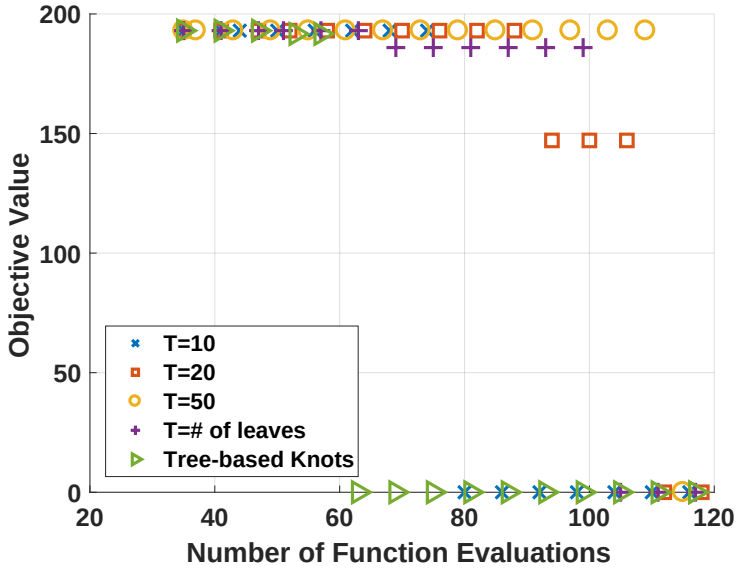


Figure 4.7: TK-MARS vs. Original MARS with different number of eligible knot locations – Sphere function

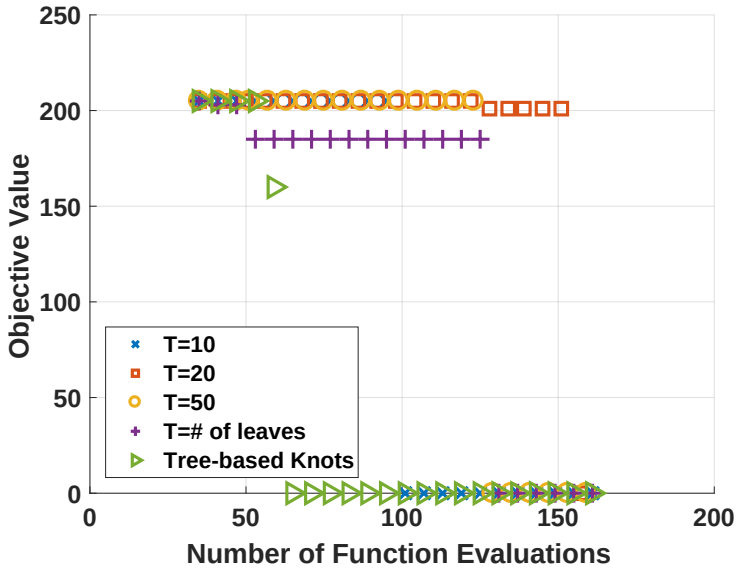


Figure 4.8: TK-MARS vs. Original MARS with different number of eligible knot locations – Levy function

4.4.2 TK-MARS vs. RBF

In this section, our focus is to study the capability of TK-MARS in identifying the unimportant variables with different parameter settings. We compare the performance of TK-MARS versus RBF using different numbers of important variables in the underlying black-box function. We consider four levels, 1, $\frac{3}{4}$, $\frac{1}{2}$, and $\frac{1}{4}$, for the fraction of important variables, which we will denote as fiv.

We perform two separate full factorial designs; one on three the parameters of TK-MARS in Table 4.2, $maxIA$, $Alg3$, and, $Robust$, and the other on the settings of RBF presented in Table 4.3. The results are in Tables 4.4 and 4.5. In Table 4.4, the first three columns correspond to important parameters of MARS: Maximum number interactions allowed (1, 2 and 3), the robust MARS option (1: on, 0: off), and the backward elimination option (1: on, 0: off). The remaining columns denote the fraction of important variables considered in computer experiments (recall from Section 3.2). Under each level of the fraction of important variables, we see two cells. The first one corresponds to Area Under the Curve (AUC) and the second value shows the total number of function evaluations before the optimum solution is found. For this set of experiments, we intentionally add the global optimum of the test functions in the fixed pool R to study the performance of TK-MARS in identifying it. Later, in the comprehensive study, Section 4.5, we use a dynamic pool generation to converge to a global optimum.

In Table 4.5, the columns show the fraction of important variables considered in computer experiments. We perform the experiments for different RBF settings and different test functions as in Table 4.5.

Table 4.4: TK-MARS Results

	$maxIA$	$Robust$	$Alg3$	**1**		**3/4**		**1/2**		**1/4**	
Rosenbrock	1	1	1	0.0203	59	0.0145	53	0.0229	62	0.0162	323
	1	1	0	0.0203	59	0.0251	64	0.0116	50	0.0204	65
	1	0	1	NA							
	1	0	0								
	3	1	1	0.1071	149	0.0772	118	0.0652	110	0.0273	209
	3	1	0	0.0927	134	0.0763	117	0.0505	91	0.0160	119
	3	0	1	0.0289	77	0.1062	148	0.0396	79	0.0198	119
	3	0	0	0.0261	65	0.0589	99	0.0666	107	0.0098	70
Rastrigin	1	1	1	0.0376	77	0.0415	81	0.0087	47	0.0522	93
	1	1	0	0.0145	53	0.0029	41	0.0372	83	0.0338	73
	1	0	1	NA							
	1	0	0								
	3	1	1	0.1476	193	0.2178	266	0.1535	197	0.1014	146
	3	1	0	0.2398	288	0.0888	130	0.0550	95	0.1464	198
	3	0	1	0.4497	518	0.1998	245	0.0309	165	0.1260	171
	3	0	0	0.3407	391	0.3069	356	0.3875	469	0.5818	708
Sphere	1	1	1	0.0193	59	0.0203	59	0.0076	47	0.0087	47
	1	1	0	0.0087	47	0.0087	47	0.0076	47	0.0067	47
	1	0	1	NA							
	1	0	0								
	3	1	1	0.0135	52	0.0193	58	0.0368	77	0.0087	47
	3	1	0	0.0145	53	0.0350	77	0.0068	45	0.0029	41
	3	0	1	0.0203	59	0.0434	83	0.0453	87	0.0125	51
	3	0	0	0.0295	71	0.0669	122	0.0607	103	0.0125	51
Levy	1	1	1	0.0248	65	0.0460	95	0.0087	47	0.0345	97
	1	1	0	0.0145	53	0.0248	68	0.0162	55	0.0135	52
	1	0	1	NA							
	1	0	0								
	3	1	1	0.0774	125	0.1500	238	0.0275	104	0.0235	65
	3	1	0	0.1640	217	0.1236	166	0.0148	55	0.0145	53
	3	0	1	0.0647	105	0.1665	241	0.0916	151	0.0222	61
	3	0	0	0.0029	41	0.2932	424	0.1719	274	0.0145	53

Table 4.5: RBF Results

	MQ-0							
	1		**3/4**		**1/2**		**1/4**	
Rosenbrock	0.0087	47	0.0077	46	0.0126	52	0.0203	131
Rastrigin	0.0087	47	0.0087	47	0.0145	53	0.0666	107
Sphere	0.0087	47	0.0087	47	0.0079	47	0.0203	59
Levy	0.0145	53	0.0145	53	0.0203	59	0.0366	83
	G-0							
	1		**3/4**		**1/2**		**1/4**	
Rosenbrock	0.7376	895	0.7156	895	0.4293	895	0.2935	896
Rastrigin	0.0087	47	0.0087	47	0.0087	47	0.0087	47
Sphere	0.0083	47	0.0074	47	0.0087	47	0.0145	53
Levy	0.8338	1035	0.8350	1035	0.6128	1035	0.4628	1035
	C-1							
	1		**3/4**		**1/2**		**1/4**	
Rosenbrock	0.0183	57	0.0219	65	0.0287	69	0.0434	309
Rastrigin	0.0241	63	0.0319	71	0.0550	95	0.3304	397
Sphere	0.0193	58	0.0183	57	0.0203	59	0.0376	77
Levy	0.0241	63	0.0261	65	0.0425	82	0.0531	93
	TPS-1							
	1		**3/4**		**1/2**		**1/4**	
Rosenbrock	0.0225	62	0.0219	65	0.0361	78	0.0245	171
Rastrigin	0.0245	61	0.0261	65	0.0290	68	0.0705	111
Sphere	0.0241	63	0.0241	63	0.0193	58	0.0319	71
Levy	0.0357	75	0.0424	83	0.0338	73	0.0666	107

For instance, in Table 4.4, the first row corresponds to the result of TK-MARS on the Rosenbrock function with no interactions ($maxIA = 1$), using the robust version ($Robust = 1$) and backward elimination ($Alg3 = 1$). Looking at the first column where all of the variables are important, 0.0203 and 59 denote the AUC measure and the total number of function evaluations before the optimum solution found, respectively. Generally, a smaller AUC and a smaller number of function evaluations are of interest. The explained cell can be compared with the counterpart cells in Table 4.5; the first row in each stack shows different settings of RBF. For MQ-0, the corresponding values for Rosenbrock function with all variables being important are 0.0087 and 47, for G-0, they are 0.7376 and 895, for C-1, they are 0.0183 and 57, and for TPS-1, they are 0.0225 and 62. In this specific case, we can say TK-MARS with no interaction ($maxIA = 1$) outperforms RBF with Gaussian and Thin Plate Spline bases.

For simplicity, let us consider the TK-MARS setting in the form of a vector ($maxIA$, $Robust$, $Alg3$). For instance, (3,0,1) corresponds to $maxIA = 3$, $Robust = 1$ (on) and $Alg3 = 0$ (off). For the fraction of important variables, we will continue with $fiv = 1$ when all of the variables are important and $fiv = \frac{3}{4}$, $fiv = \frac{1}{2}$, and $fiv = \frac{1}{4}$ for the other levels. Below, we discuss the conclusion from the experiments for each of the test functions across different levels of the fraction of important variables.

Rosenbrock: From Table 4.4, note that the best result for $fiv = 1$, corresponds to $(1, 1, 1)$ and $(1, 1, 0)$ with 59 total number of function evaluations and $AUC = 0.0203$. Rosenbrock has interactions, however, TK-MARS finds the opti-

mum faster considering no interactions. That is because the Rosenbrock function only contains $x_{i+1}x_i^2$ interaction terms which are relatively small to be captured, as in Table 4.1-Row 1. We can justify this by the unnecessary interaction basis functions that appear when interactions are allowed in MARS. Hence, TK-MARS is overparameterized for Rosenbrock, which worsens the optimization results.

Dropping the number of considered important variables to 20 ($\frac{3}{4} * 27 = 20$), the best result goes to $(1, 1, 0)$ again with 53 total number of function evaluations. In this case, we can say TK-MARS is successful in identifying the unimportant variables and determines the structure of the underlying function to find the optimum very well. For $\mathit{fiv} = \frac{1}{2}$ and $\mathit{fiv} = \frac{1}{4}$, the performance of TK-MARS maintains strong with the selected setting of $(1, 1, 0)$.

Rastrigin: The best result corresponds to $(1, 1, 0)$ with the total number of function evaluations equaling 53 and $AUC = 0.0145$. The Rastrigin function has multiple local optima. However, TK-MARS (a regressive model) is successful in capturing the overall pattern. This function does not have any interactions. Consequently, TK-MARS with no interactions performs better. Decreasing fiv, the quality of the performance of TK-MARS maintains strong with $(1, 1, 0)$ as the selected set of parameters.

Sphere: The best result corresponds to $(1, 1, 0)$ with the total number of function evaluations equals to 47 and $AUC = 0.0087$. The Sphere function is a well-structured function without any interactions. For the Sphere function the performance of TK-MARS for optimization is notable. However, the best performing set of parameters changes from $(1, 1, 0)$ to $(3, 1, 0)$ for $\mathit{fiv} = \frac{1}{2}$ and $\mathit{fiv} = \frac{1}{4}$, re-

spectively. It is worth mentioning that the performance of the $(1, 1, 0)$ setting is slightly worse than other settings but is still a good set of parameters for optimization.

Levy: The best result corresponds to $(3, 0, 0)$ with the number of evaluations equals to 41. The Levy function has multiple local optima and is highly nonconvex without any prevailing shape. Although, it does not have any interactions, TK-MARS tries to capture the structure of Levy, including the interaction terms and no backward elimination. As a result, TK-MARS brings in as many effective basis functions as it can to find the optimum faster. Hence, $(3, 0, 0)$ is selected. It has not escaped our notice that the performance of $(1, 1, 0)$ setting of TK-MARS is slightly worse than $(3, 0, 0)$ and can be chosen as a good setting for Levy function.

We can say that TK-MARS works remarkably better for the Levy function when there are fewer important variables. The best setting of parameters are $(1, 1, 0)$ for $\mathit{fiv} = \frac{1}{2}$ and $(1, 1, 1)$ for $\mathit{fiv} = \frac{1}{4}$. However, for this case, $(1, 1, 0)$ can be considered as a good setting as well.

In general, the selected parameter setting for the black-box optimization with TK-MARS is chosen to be $(1, 1, 0)$. This means that TK-MARS without any complex structure and overparameterizing can find the prevailing shape of the underlying function appropriately for optimization. Researchers are very focused on interpolation in the surrogate optimization literature, which may in fact be inferior to non-interpolating methods.

We can conclude that TK-MARS is competitive with RBF as in Figures 4.9-4.24. TK-MARS outperforms RBF with TPS basis in all of the test functions.

RBF with Cubic and Multiquadric basis, outperforms TK-MARS for Rosenbrock and Rastrigin. Besides, RBF with Gausian basis slightly outperforms TK-MARS for the Rastrigin function. Please note that the results are provided for a single run, as our objective is to find the best setting. Later in Section 4.5 we shall provide a comprehensive comparison for multiple runs.

Another aspect of the results in Figures 4.9-4.24 is the consistent quality of the performance of TK-MARS maintains through different levels of fiv. However, it is not the case for RBF. The performance of RBF for optimization decays as the number of considered important variables decays. Generally speaking, TK-MARS is more successful in identifying the important variables than RBF. As the number of important variables decreases TK-MARS outperforms RBF, significantly.

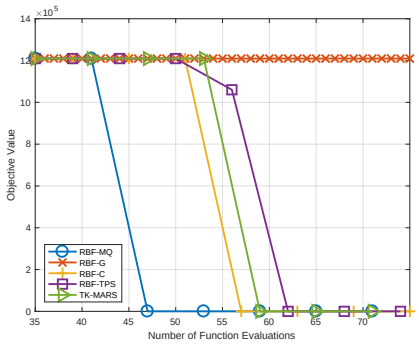


Figure 4.9: Tree-based MARS Versus RBF for Rosenbrock Function with all variables important

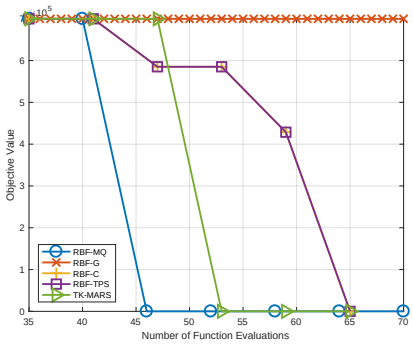


Figure 4.10: Tree-based MARS Versus RBF for Rosenbrock Function with $\frac{3}{4}$ of variables important

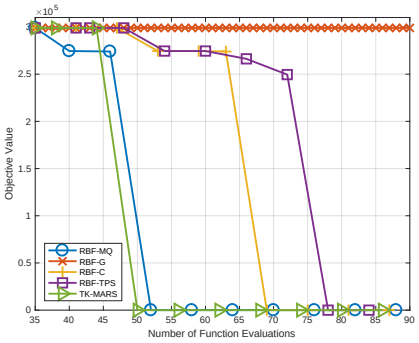


Figure 4.11: Tree-based MARS Versus RBF for Rosenbrock Function with $\frac{1}{2}$ of variables important

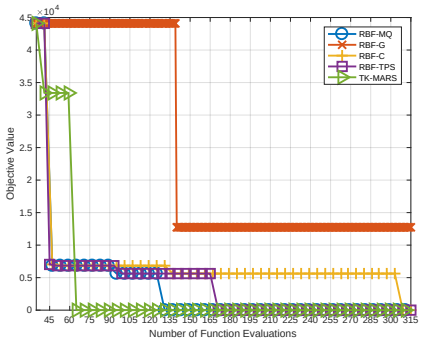


Figure 4.12: Tree-based MARS Versus RBF for Rosenbrock Function with $\frac{1}{4}$ of variables important

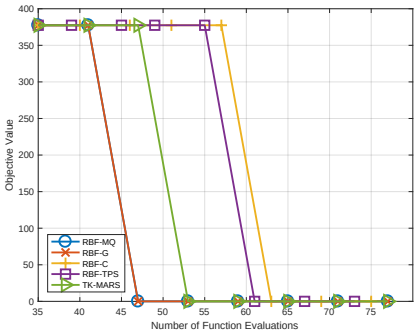


Figure 4.13: Tree-based MARS Versus RBF for Rastrigin Function with all variables important

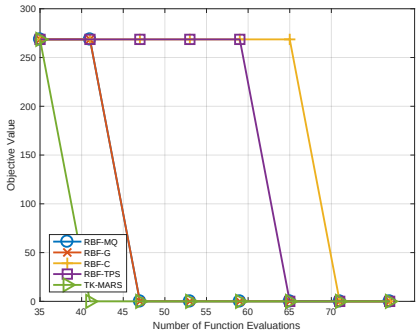


Figure 4.14: Tree-based MARS Versus RBF for Rastrigin Function with $\frac{3}{4}$ of variables important

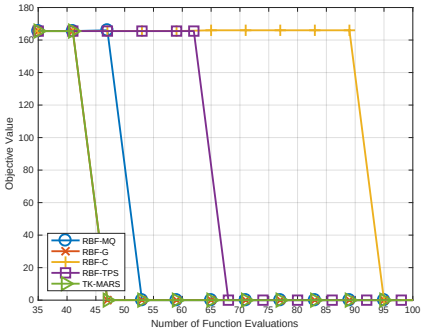


Figure 4.15: Tree-based MARS Versus RBF for Rastrigin Function with $\frac{1}{2}$ of variables important

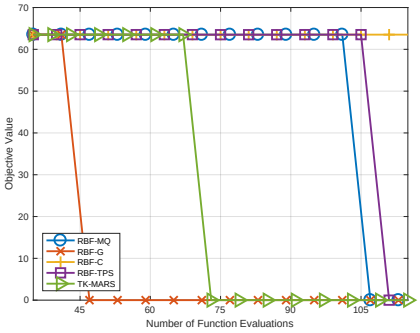


Figure 4.16: Tree-based MARS Versus RBF for Rastrigin Function with $\frac{1}{4}$ of variables important

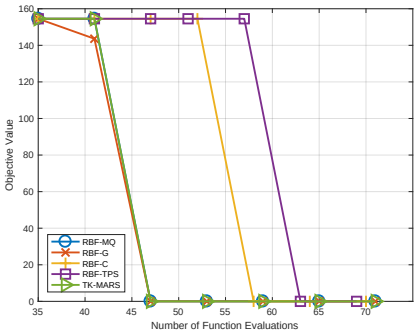


Figure 4.17: Tree-based MARS Versus RBF for Sphere Function with all variables important

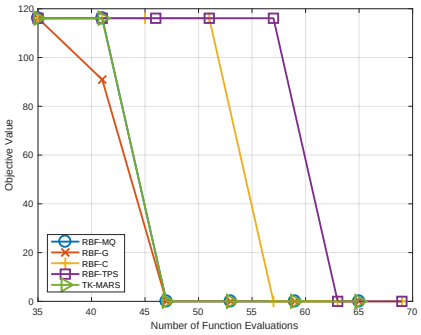


Figure 4.18: Tree-based MARS Versus RBF for Sphere Function with $\frac{3}{4}$ of variables important

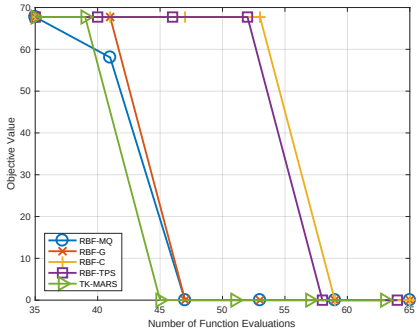


Figure 4.19: Tree-based MARS Versus RBF for Sphere Function with $\frac{1}{2}$ of variables important

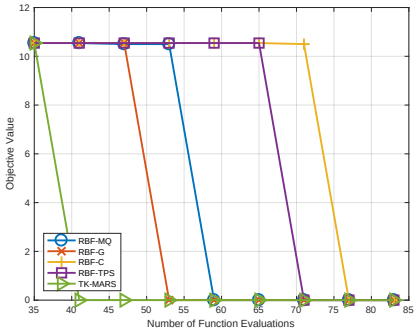


Figure 4.20: Tree-based MARS Versus RBF for Sphere Function with $\frac{1}{4}$ of variables important

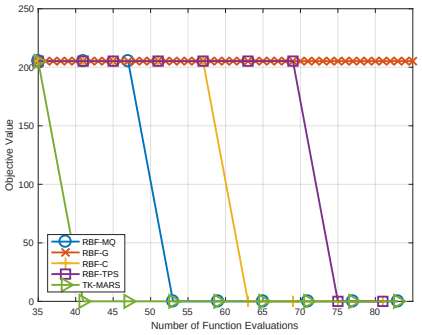


Figure 4.21: Tree-based MARS Versus RBF for Levy Function with all variables important

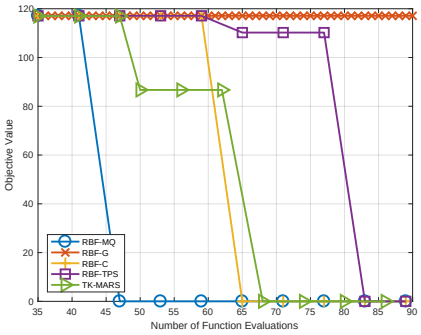


Figure 4.22: Tree-based MARS Versus RBF for Levy Function with $\frac{3}{4}$ of variables important

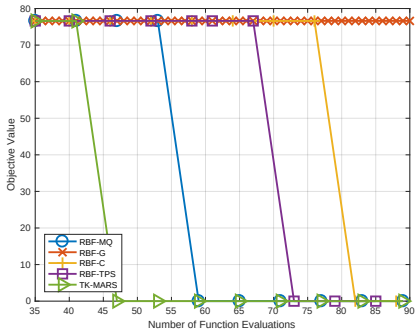


Figure 4.23: Tree-based MARS Versus RBF for Levy Function with $\frac{2}{4}$ of variables important

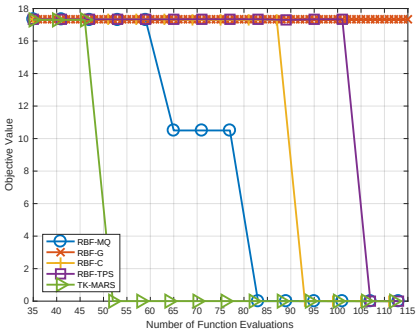


Figure 4.24: Tree-based MARS Versus RBF for Levy Function with $\frac{1}{4}$ of variables important

4.5 Comprehensive Study

In this section, we present a comprehensive study of the performance of the proposed algorithms on a new class of test functions that include unimportant variables and uncertainty. Different problem setting parameters may affect the perfor-

mance of the proposed methods. In addition, each algorithm has its own parameters that needs to be tuned. A comprehensive study requires considering different combinations of the parameters.

4.5.1 Problem setting parameters

In this research, we study a set of well-known test problems to see the performance of the proposed algorithm, especially in high-dimensional space. However, we believe that the existing test functions that are designed for global optimization, are different from the less-symmetric and less-structured real-world applications. Hence, we design a new class of test functions that mimic the real-world problems in the context of surrogate optimization [57–59].

To design a new class of test functions, we consider two major factors of real-world black-box functions, (1) fraction of important variables and (2) uncertainties associated with the black-box systems. For the first factor, in order to show that the proposed method is able to recognize the important variables among all the input variable set, the function evaluation is based on a fraction of input variables, not all of them. In addition, to represent stochastic black-box systems, an uncertainty component is added to the true objective value for each function evaluation. An initial experiment is presented in this study by adding a Gaussian noise $\tilde{f}(x) = f(x)+\epsilon$, where $\epsilon \sim N(f(x), np*range(f(x_{x\in I_o})))$, where np is the noise percentage level.

4.5.2 Algorithm parameters

Some of the parameters related to the method proposed in Chapter 3 might have a significant effect on the performance of the algorithm. In addition to the problem setting parameters, we consider these parameters for computational experiments, as well. In each experiment, we indicate the algorithm-related parameters and their different levels to be studied. The measure of performance applied is MTFAUC introduced in Section 3.7.2. Besides, some parameters related to the methods are fixed for the experimental runs.

4.5.3 Design of Experiments

To investigate the effect of the parameters on the performance of the algorithm, we need to determine a Design of Experiments (DOE) to execute organized and effective sets of experiments and collect a comprehensive set of observations. A proper design of experiments helps to determine the variations of the response caused by the key parameters. To determine the significant parameters and their impacts, an efficient design of experiments is required to collect the data. Conducting Analysis of Variance (ANOVA), we can potentially uncover the variability that is explained with different types of parameters. We can recommend a solution or an approach regarding the overall investigation based on a reliable DOE observation.

4.5.3.1 Orthogonal Array

A full factorial analysis may not be a reasonable design regarding a large set of parameters. Considering a subset of combinations of multiple parameters at multiple levels, a specific type of design that ensures all levels of all factors are considered equally is important. One of the modern types of fractional factorial design is Orthogonal Arrays (OA), which distributes design points uniformly in the design region. There are several techniques to construct orthogonal arrays with multiple factor levels [60,61]. OA designs are common for design of experiments in industries where there is a large number of factors to be studied but only a few of them effect the output. OA designs save runs and are flexible for combinations of factor levels. SAS provides a complete library of Orthogonal Arrays with different numbers of runs, factors, and levels [62].

In this research, we apply OA design to recognize the main effects of the problem setting parameters and algorithm-related parameters. There are two separate OA designs; one for the deterministic approach (no-replication) and one for the stochastic approach.

4.5.3.1.1 No-Replication Results

To examine the main effects of the parameters on the performance of the deterministic algorithm and the effect of different models, an OA design is considered on the set of parameters presented in Table 4.6. Appendix Table 1 shows the OA design that is chosen for this study.

Table 4.6: No-Replication parameters and levels for OA design

Problem Parameters	**levels**
Test function	Rosenbrock, Rastrigin, Levy
Dimension	10, 20, 30
Fraction of important variables	0.25, 0.50, 0.75, 1
Noise level (%)	0, 10, 25
Algorithm Parameters	**levels**
Initial pool size	$d+1, 2(d+1)$
DOE method	LHD, Sobol
EEPA distance	Euclidean, Cosine
EEPA # candidates	3, 6
Model	RBF, TK-MARS

Conducting an analysis of variance (ANOVA) in R, the hypothesis of variations caused by different factors is tested. In ANOVA, the reference population is the first level of each factor. From the results in Table 4.7, observe that the fraction of important variables and noise level are statistically significant parameters. Consequently, we compare each level with the reference level. The results indicate that the noise effect cannot be handled by the deterministic approach and has a significant effect on the MTFAUC value. Further, from the model factor, observe that TK-MARS is slightly better than RBF. A more precise analysis of the performance of RBF and TK-MARS is presented later with a Full-Factorial analysis in Section 4.5.3.2.

As a result of the conducted ANOVA, the deterministic approach is not able to handle the noise effect and an alternative approach should be considered to mitigate it.

Table 4.7: No-Replication ANOVA table

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	0.0213	0.06535	0.326	0.745659	
fiv=0.75	0.12197	0.04773	2.556	0.013294	*
fiv=0.50	0.22659	0.04773	4.748	1.43E-05	***
fiv=0.25	0.30299	0.04773	6.348	3.84E-08	***
noise level=10%	0.14558	0.04133	3.522	0.000851	***
noise level=25%	0.23302	0.04133	5.638	5.59E-07	***
Rastrigin	0.33031	0.04133	7.991	7.10E-11	***
Levy	-0.02759	0.04133	-0.667	0.507163	
dimension=20	-0.06038	0.04133	-1.461	0.149573	
dimension=30	0.01005	0.04133	0.243	0.808709	
poolSize=2(d+1)	0.01179	0.03375	0.35	0.728	
DOE=Sobol	-0.02552	0.03375	-0.756	0.452732	
EEPA distance=Cosine	0.01675	0.03375	0.496	0.62165	
EEPA number of candidates=6	0.0452	0.03375	1.339	0.185774	
model=TK-MARS	-0.01327	0.03375	-0.393	0.695633	
Signif. Codes:	0 '***'	0.001 '**'	0.01 '*'	0.05 '.'	0.1 ' ' 1

4.5.3.1.2 Replication Results

In this section, we consider the proposed replication strategies to mitigate the noise effect. An analysis of variance is conducted on a set of parameters shown in Table 4.8 with MTFAUC as the response variable. We consider two replication types of *fixedrep* and $smartrep$ with two different replication numbers of 5 and 10. For *fixedrep* the replication number is the fixed number of replications for each candidate points, however, for $smartrep$ the replication number is the maximum number of replications. We use $fixedrep, 5$, $fixedrep, 10$, $smartrep, 5$, and $smartrep, 10$ for different replication types with different number of replications. The OA design chosen for this study is presented in Appendix Table 2.

From Table 4.9, observe that the noise parameter is not significant anymore, which indicates that replication helps to cancel out the noise effect. TK-MARS is

Table 4.8: W/Replication parameters and levels for OA design

Problem Parameters	**levels**
Test function	Rosenbrock, Rastrigin, Levy
Dimension	10, 20, 30
Fraction of important variables	0.25, 0.50, 0.75, 1
Noise level (%)	5, 10, 25
Algorithm Parameters	**levels**
Initial pool size	$d+1$, $2(d+1)$
DOE method	LHD, sobol
EEPA distance	Euclidean, Cosine
EEPA # candidates	3, 6
Replication type	fixed_rep, smart_rep
Replication #	5, 10
Model	RBF_avrg, TK-MARS_avrg, TK-MARS_keepall

marginally significant. It is worth mentioning that the Smart-Replication strategy is statistically better than Fixed-Replication, which is consistent with our expectation. Using smart replication saves function evaluations and reduces the MTFAUC in the noisy situation.

Table 4.9: Replication ANOVA table

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	0.250806	0.086891	2.886	0.005626	**
fiv=0.75	0.07965	0.056382	1.413	0.163596	
fiv=0.50	0.207822	0.056382	3.686	0.000537	***
fiv=0.25	0.132775	0.056382	2.355	0.022263	*
model=TK-MARS_avrg	-0.04463	0.048828	-0.914	0.364833	
model=TK-MARS_keepall	0.113446	0.048828	2.323	0.024028	*
noise level=10%	0.004939	0.048828	0.101	0.91982	
noise level=25%	0.009702	0.048828	0.199	0.843259	
smart_rep	-0.09262	0.039868	-2.323	0.02404	*
Rastrigin	0.344011	0.048828	7.045	3.81E-09	***
Levy	0.045359	0.048828	0.929	0.357128	
dimension=20	0.073677	0.048828	1.509	0.137262	
dimension=30	0.051091	0.048828	1.046	0.300151	
Replication \#=10	0.146338	0.048828	2.997	0.004142	**
poolSize=2(d+1)	0.016954	0.039868	0.425	0.67238	
DOE=Sobol	-0.00669	0.039868	-0.168	0.867332	
EEPA distance=Cosine	-0.01711	0.039868	-0.429	0.669465	
EEPA number of candidates=6	-0.02138	0.039868	-0.536	0.594059	
Signif. Codes:	0 '***'	0.001 '**'	0.01 '*'	0.05 '.'	0.1 ' ' 1

4.5.3.2 Full Factorial

Identifying the key parameters of the OA design results from Section 4.5.3.1.2, we study a full model analysis on the significant problem and algorithm parameters including the main effects and the interaction effects.

Similar to the OA design, we have two separate designs for No-Replication and Replication strategies. Looking at the ANOVA tables from the OA design, Tables 4.6 and 4.8, we drop the unimportant parameters to investigate the variation caused by important parameters on the MTFAUC. As a consequence, dimension, initial pool size, the DOE method used to design the initial pool, the EEPA distance metric, and the EEPA number of candidate points factors are not statistically significant. Hence, to conduct a Full-Factorial design, these parameters are fixed to the significant level based on OA ANOVA. We will continue with TK-MARS with averaging over replicated points and ignore the version that keeps all the replicated points since ANOVA shows there is no advantage in using the latter.

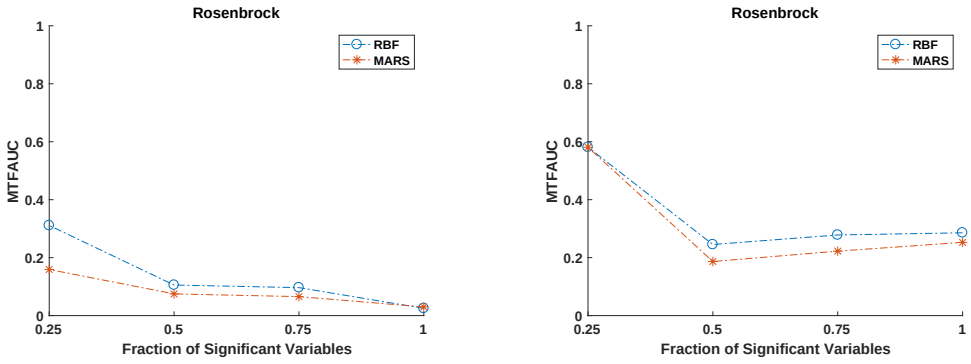
4.5.3.2.1 No-Replication Results

In this section, we study the effect of different surrogate models employed and the fraction of important variables. From the ANOVA on the OA design, Table 4.6, TK-MARS is not statistically but slightly significant. A Full-Factorial design focusing on the surrogate model used determines the impact of the models applied to the response. Table 4.10 represents the parameters to be studied and the fixed parameters for collecting the observations of Full-Factorial design.

Table 4.10: No-Replication parameters and levels for Full-Factorial design

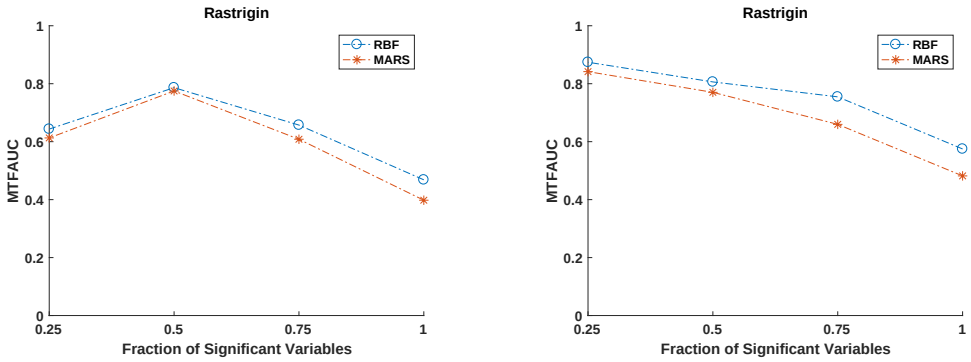
Problem Parameter	**levels**
Test function	Rosenbrock, Rastrigin, Levy
Dimension	fixed=30
Fraction of important variables	0.25, 0.50, 0.75, 1
Noise level (%)	0, 25
Algorithm Parameters	**levels**
Initial pool size	fixed=d+1
DOE method	fixed=LHD
EEPA distance	fixed=Euclidean
EEPA # candidates	fixed=3
Model	RBF, TK-MARS

Before conducting an ANOVA, looking into the plots of different models used and their impact under different noise levels and fraction of important variables is worthwhile. Figures 4.25a-4.27b correspond to each test functions in two different cases: (a) no noise and (b) high noise.



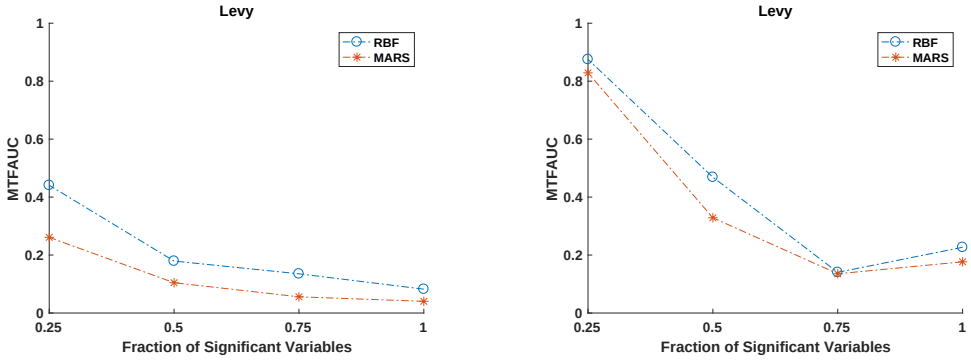
(a) No noise (b) High noise

Figure 4.25: TK-MARS vs. RBF performance for Rosenbrock function



(a) No noise (b) High noise

Figure 4.26: TK-MARS vs. RBF performance for Rastrigin function



(a) No noise (b) High noise

Figure 4.27: TK-MARS vs. RBF performance for Levy function

Note that as the fraction of important variables decreases, MTFAUC increases. It is harder for the algorithm to identify important variables when fewer variables are used to evaluate the output.

From Table **??**, based on the additive model, TK-MARS statistically outperforms RBF at the significance level of $\alpha = 0.1$. We see that the noise effect has

not been mitigated in this approach. These results are consistent with the results of No-Replication in the OA design, Table 4.7.

$model*fraction$ interaction effect is not significant in the full model ANOVA, Table 4.12. This is also consistent with our observations in the plots, Figures 4.25a-4.27b. However, there is a slight $noise * fraction$ interaction. This shows that the effect of noise differs, depending on the fraction of important variables. In the plots, we see that moving from no noise (Figure 4.27a) to high noise (Figure 4.27b), there is a slight shift in the patterns.

Table 4.11: No-Replication ANOVA additive model on Full-Factorial design

	Estimate	Std. Error	t value	Pr(>|t|)		
(Intercept)	0.34955	0.04634	7.543	3.32E-09	***	
Rastrigin	0.45076	0.04013	11.232	6.10E-14	***	
Rlevy	0.06121	0.04013	1.525	0.13507		
noise level=25%	0.18583	0.03277	5.671	1.37E-06	***	
fiv=0.50	-0.18157	0.04634	-3.918	0.00034	***	
fiv=0.75	-0.26667	0.04634	-5.755	1.05E-06	***	
fiv=1	-0.3306	0.04634	-7.134	1.22E-08	***	
model=TK-MARS	-0.05796	0.03277	-1.769	0.08453	.	
Signif. Codes:	0 '***'	0.001 '**'	0.01 '*'	0.05 '.'	0.1 ' '	1

Table 4.12: No-Replication ANOVA full model on full-factorial design

	Estimate	Std. Error	t value	Pr(>|t|)		
(Intercept)	0.35733	0.05643	6.332	2.23E-07	***	
Rastrigin	0.45076	0.04168	10.815	5.20E-13	***	
Rlevy	0.06121	0.04168	1.469	0.1504		
noise level=25%	0.18583	0.03403	5.46	3.38E-06	***	
fiv=0.50	-0.18905	0.06806	-2.778	0.008547	**	
fiv=0.75	-0.27718	0.06806	-4.072	0.000236	***	
fiv=1	-0.34372	0.06806	-5.05	1.21E-05	***	
model=TK-MARS	-0.07352	0.06806	-1.08	0.287057		
fiv=0.50:model=TK-MARS	0.01497	0.09626	0.155	0.877273		
fiv=0.75:model=TK-MARS	0.02102	0.09626	0.218	0.828312		
fiv=1:model=TK-MARS	0.02624	0.09626	0.273	0.78669		
Signif. Codes:	0 '***'	0.001 '**'	0.01 '*'	0.05 '.'	0.1 ' '	1

4.5.3.2.2 Replication Results

In this section, we study the effect of different approaches proposed in this research at different noise levels. A Full-Factorial design focusing on the proposed methods is considered. Table 4.13 represents the parameters to be studied and the fixed parameters for collecting the observations of the Full-Factorial design. In this design, the replication type and replication number construct $3 \times 2 = 6$ levels for a single $repType$ factor in order to study the No-replication approach together with replication approaches. The test function is the blocking variable. Four different noise levels are investigated for a comprehensive study.

The fraction of important variables is statistically significant in almost all of the analysis. However, for simplicity and focusing on the noise levels, we continue with the fixed $fiv = 0.5$, which is the most significant level based on Table 4.9. Just to focus on the different approaches for handling the uncertainty effect, we separate out the TK-MARS results from the RBF results. From No-Replication, we conclude that TK-MARS is statistically better than RBF at the significance level of $\alpha = 0.1$. Now, we study how interpolating and non-interpolating models cope with replications.

- **TK-MARS: Non-Interpolating Model**

Figures 4.28, 4.31, and 4.34 show the performance of different approaches across different noise levels for different test functions. In these plots and the corresponding ANOVA, Tables 4.14-4.15, the surrogate model applied is TK-MARS. We shall later show the results for RBF as well. The plots in Figures 4.28, 4.31,

Table 4.13: W/Replication parameters and levels for Full-Factorial design

Problem Parameters	**levels**
Test function	Rosenbrock, Rastrigin, Levy
Dimension	fixed=30
Fraction of important variables	fixed=0.5
Noise level (%)	0, 5, 10, 25
Algorithm Parameters	**levels**
Initial pool size	fixed=d+1
DOE method	fixed=LHD
EEPA distance	fixed=Euclidean
EEPA # candidates	fixed=3
Replication Type	norepl, fixed_rep, smart_rep
Replication #	5, 10
Model	RBF_avrg, TK-MARS_avrg

and 4.34 show that No-Replication outperforms the Replication approaches. Observe that, Smart-Replication outperforms Fixed-Replication approaches. As the noise level increases, the MTFAUC increases as expected.

Figure 4.29, shows the box-plot of the MTFAUC values for 30 different runs on different noise levels with different methods, Figures 4.29a-4.29e. From Figure 4.28, No-Replication outperforms Replication approaches. However, looking at Figure 4.29c ($fixedrep, 10$) and Figure 4.29c ($smartrep, 10$), we confirm that the MTFAUC variance is more reasonable than $norepl$, Figure 4.29a. As in Figures 4.29d and 4.29e, in lower levels of noise, Smart-Replication have the shortest box and are competitive with the No-Replication approach, which indicates the robustness of the Smart-Replication approach to randomness. Although, $fixedrep, 10$ is robust to different noise levels and there is no significant difference in the means across the noise levels, the overall average MTFAUC is larger in lower noise levels compare to $norepl$ and $smartrep$. This is because of unnecessary function evaluations are done by the Fixed-Replication approach. In

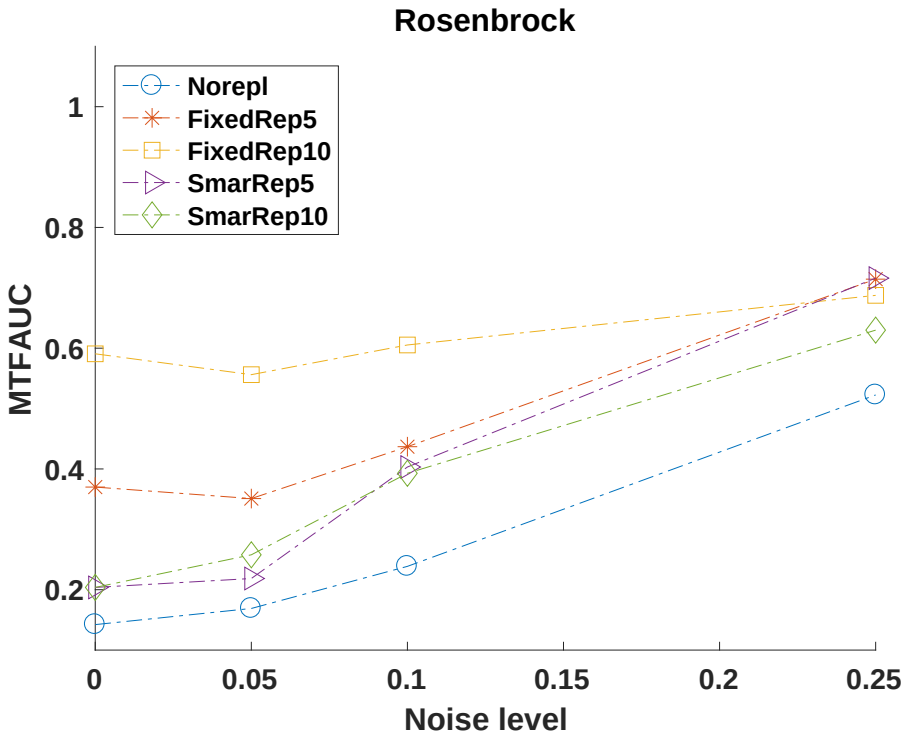


Figure 4.28: MTFAUC of different strategies with TK-MARS across different noise levels

addition, from the box-plots, note that as the level of noise increases, the variance of the No-Replication's $MTFAUC$ based on different pools increases.

From Figures 4.30, we see the variance of the objective values of BSMS after 1000 function evaluations for 30 different runs at each considered noise level. As shown in the Figure, in the no-noise case, $smartrep$ is competitive with $norepl$ and they have the shortest boxes, comparatively. $fixedrep, 10$ and $smartrep = 10$ are robust to different noise levels since the means are closer compared to $norepl$. In the highest level of noise, $smartrep, 10$ has the shortest box, indicating that

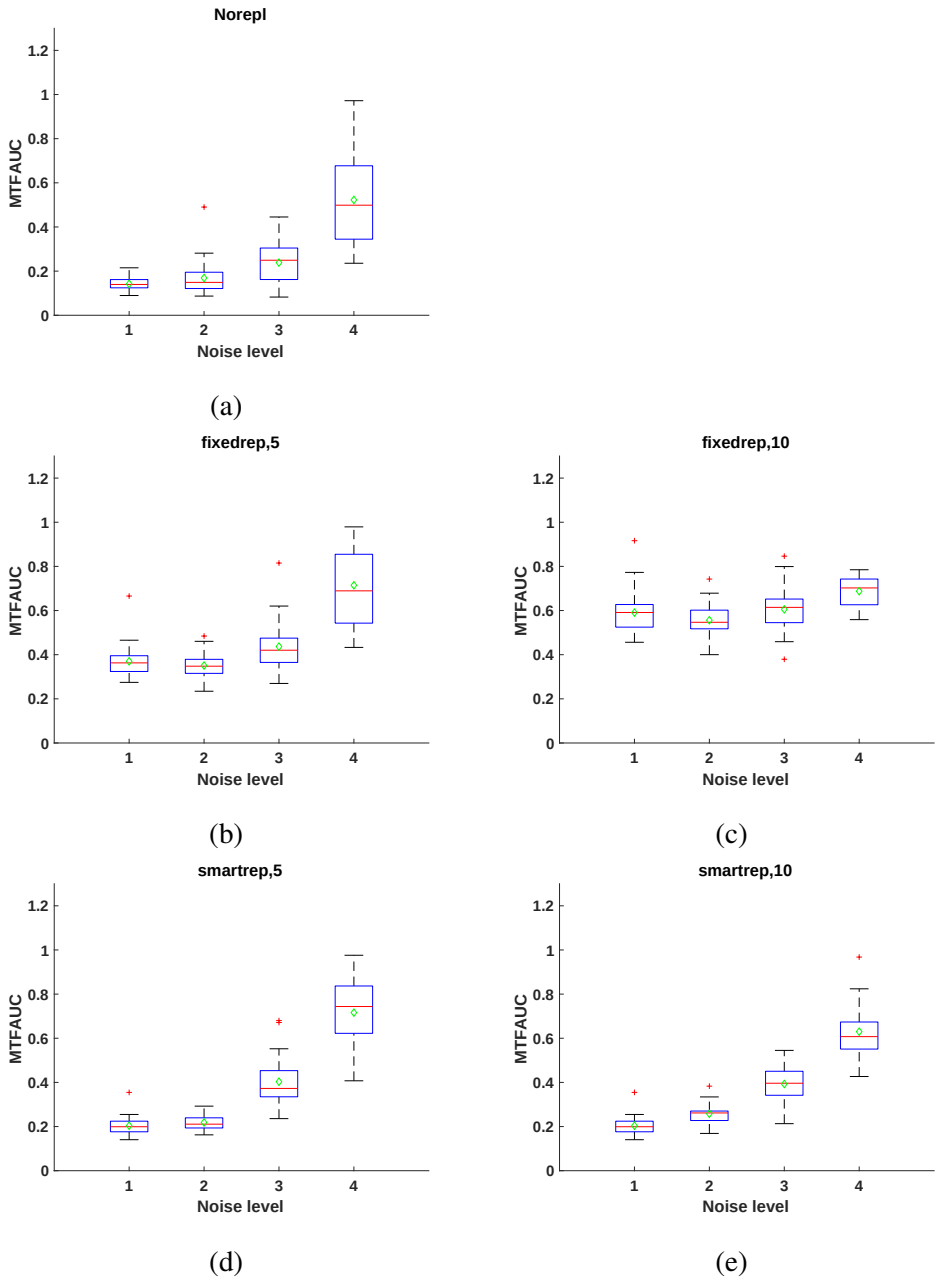


Figure 4.29: MTFAUC box-plots of Full-Factorial design results for Rosenbrock function with TK-MARS

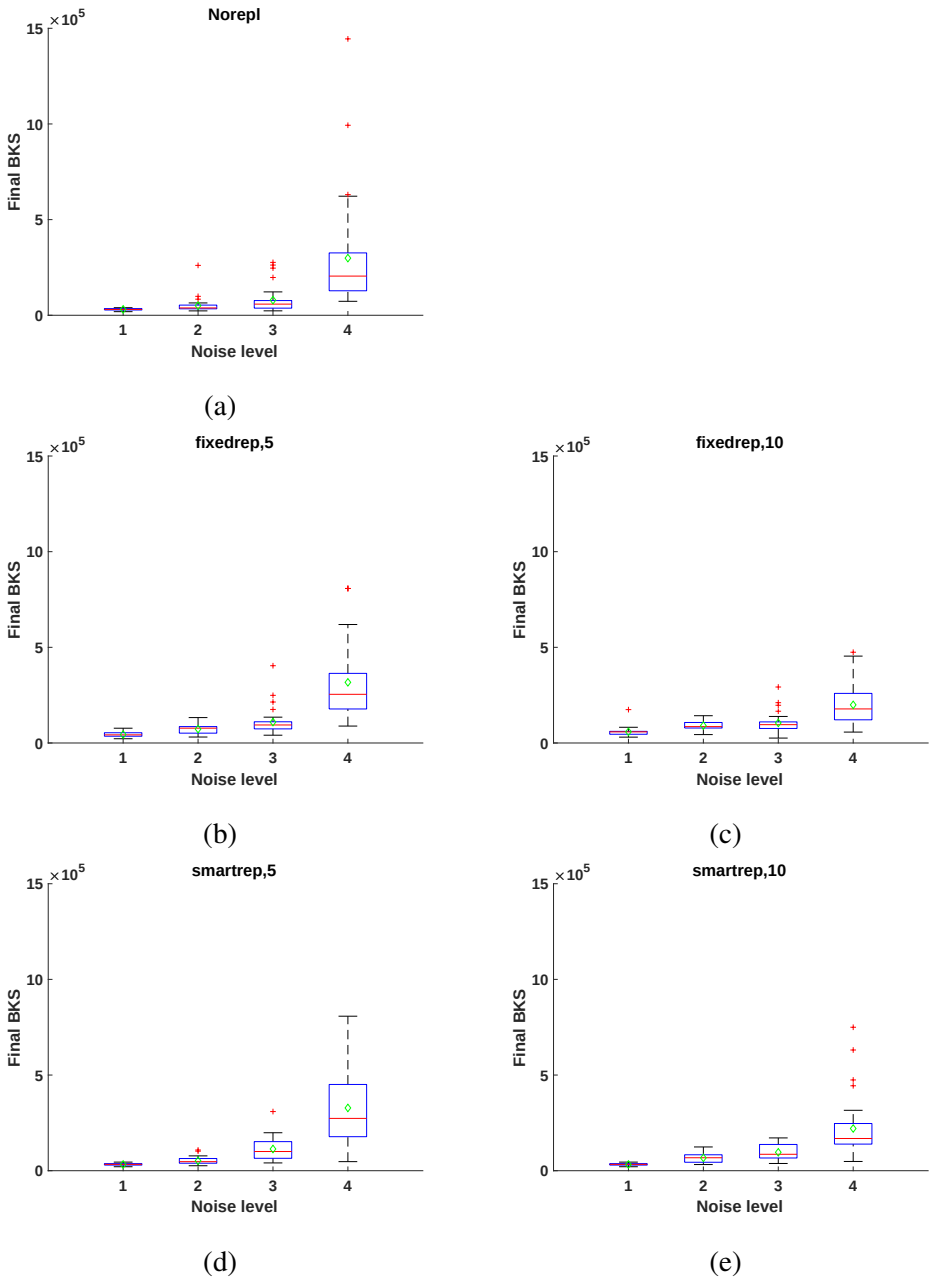
$smartrep, 10$ is more robust to randomness and is more reliable for the Rosenbrock function.

Looking into the results on the Rastrigin function, Figure 4.31, we can see that there is a small difference between the different methods. The reason can be justified by the highly fluctuating behavior of the Rastrigin function with several local optima, and as the noise level added to the function increases, the optimum is harder to obtain.

Figure 4.32 shows the box-plot of the MTFAUC values for 30 different runs on different noise levels with different methods, Figures 4.32a-4.32e. For the Rastrigin function, which is a highly fluctuating function, note that $fixedrep, 10$, which is the full replication approach, has the shortest box and is the most robust approach to randomness and across different noise levels. However, in lower noise levels, $fixeedrep, 10$ has slightly higher average MTFAUC. This shows that replication helps to mitigate the fluctuations and uncertainties associated with the black-box function.

From Figures 4.33, we see the variance of the BSMS after 1000 function evaluations for 30 different runs at each considered noise level. As we can see, $frixedrep, 10$ has the shortest box comparatively; however, it has a larger BSMS. $norepl$ finds better BSMS at the end of 1000 function evaluations when the uncertainty level is low. However, $smartrep, 10$ finds better BSMS at the highest noise level. We can see there are several outliers shown in the plots, which can be justified by the highly fluctuating behavior of the Rastrigin function.

For the Levy function, No-Replication clearly outperforms the replication ap-



(a)

(b)

(c)

(d)

(e)

Figure 4.30: Final BSMS box-plots of Full-Factorial design results for Rosenbrock function

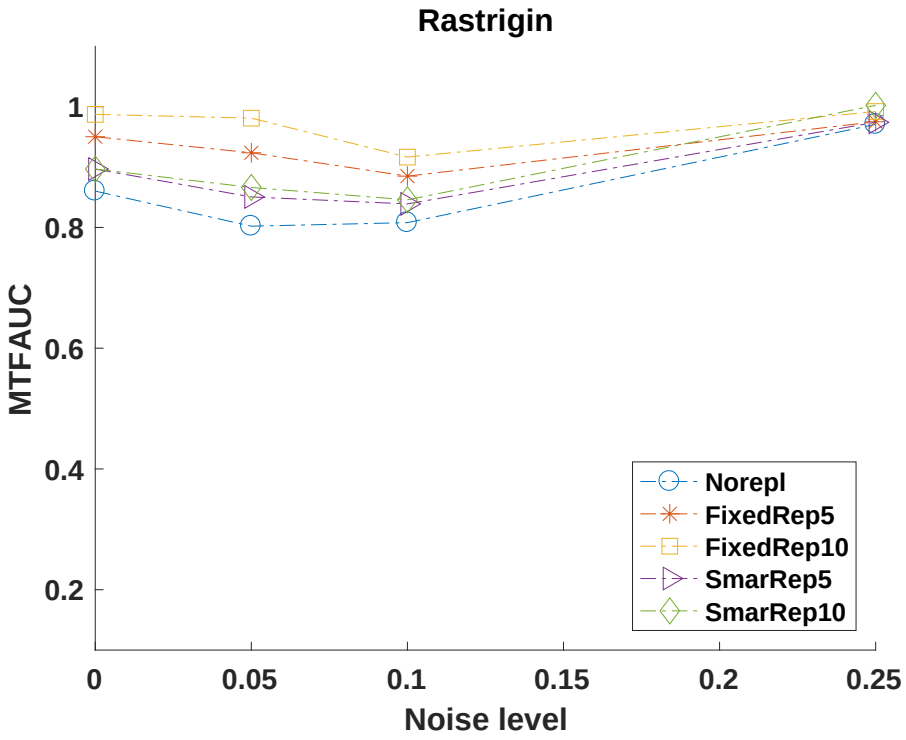


Figure 4.31: MTFAUC of different strategies with TK-MARS across different noise levels

proaches in almost every case (Figure 4.34). Although, we can see that Smart-Replication is doing well too.

Figure 4.35 shows the box-plot of the MTFAUC values for 30 different runs on different noise levels with different methods (Figures 4.35a-4.35e). From Figure 4.28, No-Replication outperforms Replication approaches. However, $smartrep, 10$ (Figure 4.29e box-plot) has comparatively short box at the higher level of noise. Overall, No-Replication has the lower average MTFAUC, but $smartrep$ is quite competitive.

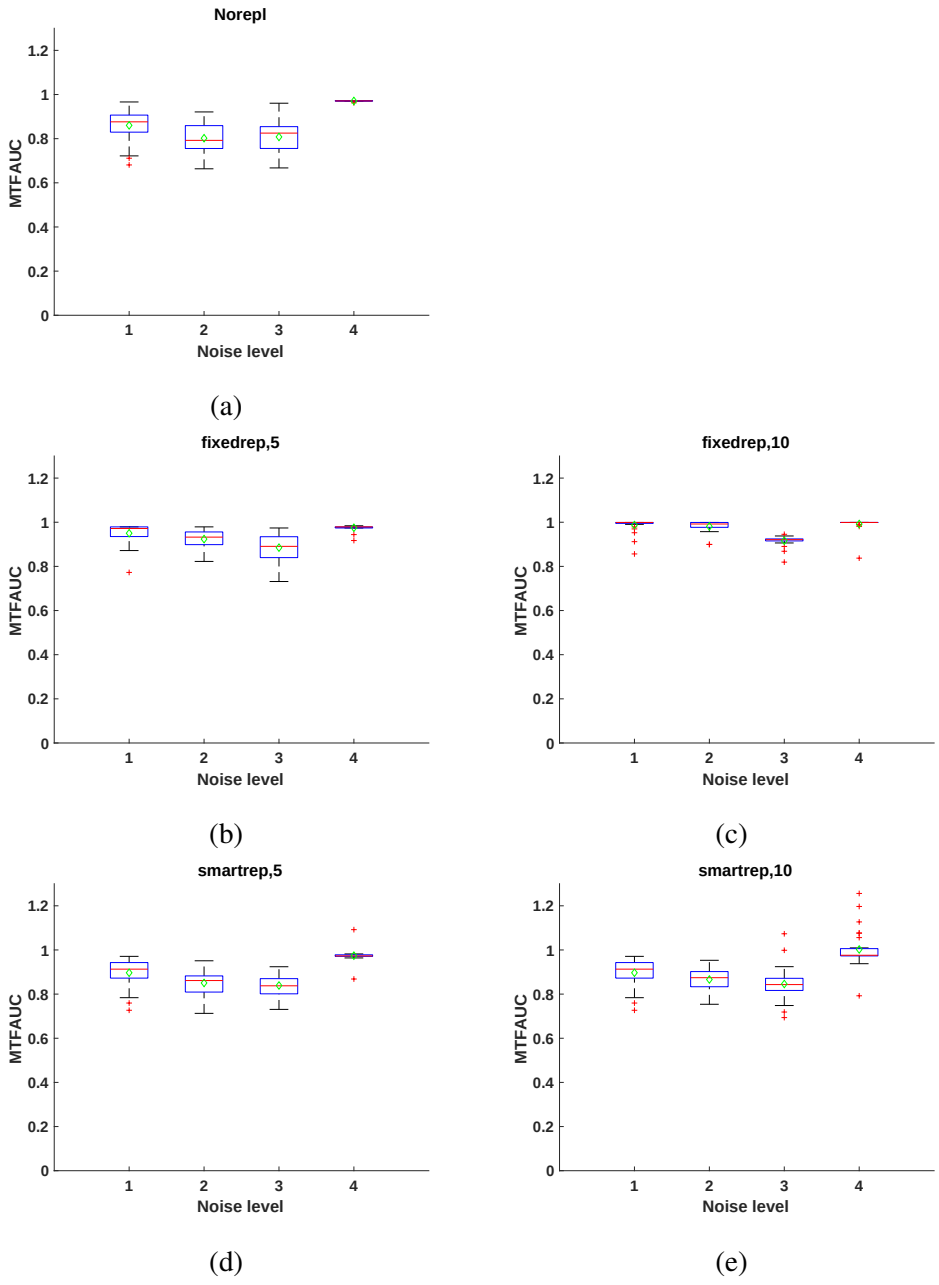
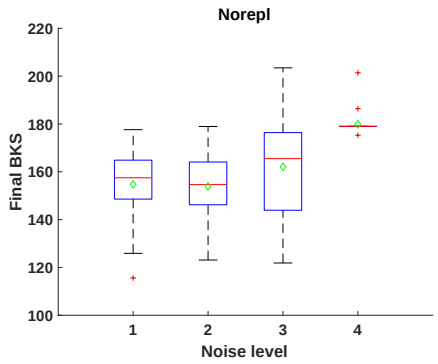
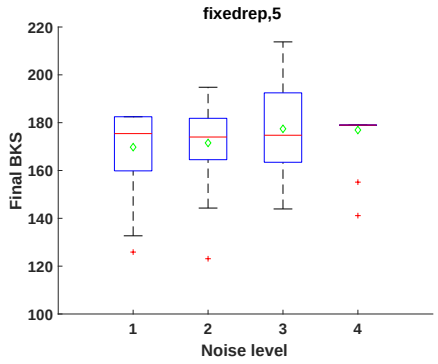


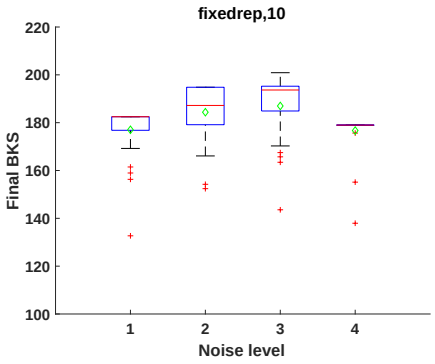
Figure 4.32: MTFAUC box-plots of Full-Factorial design results for Rastrigin function



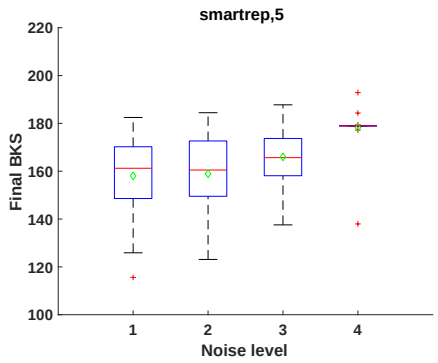
(a)



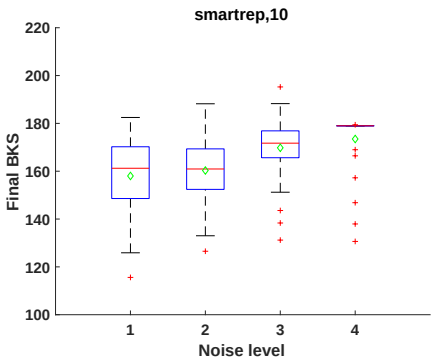
(b)



(c)



(d)



(e)

Figure 4.33: Final BSMS box-plots of Full-Factorial design results for Rastrigin function

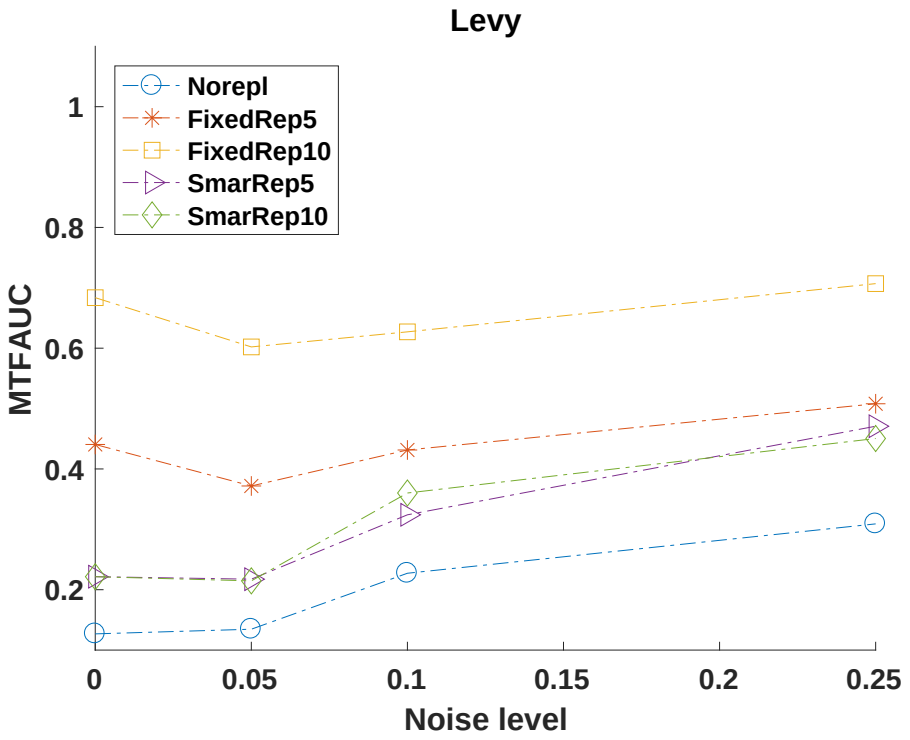
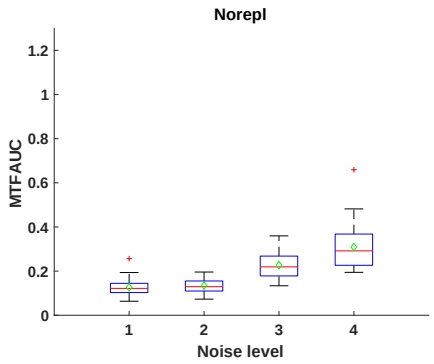


Figure 4.34: MTFAUC of different strategies with TK-MARS across different noise levels

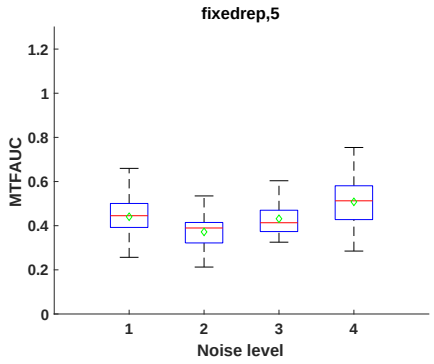
It is worth mentioning that, *fixedrep* has the highest robustness (shortest boxes) across different noise levels, although it is outperformed by *norepl* and *smartrep* in most of the cases.

The variance of the BSMS after 1000 function evaluations for 30 different runs at each considered noise level is presented in Figure 4.36. As we can see, No-Replication outperforms in almost all of the cases in terms of the BSMS after 1000 evaluation for the Levy function.

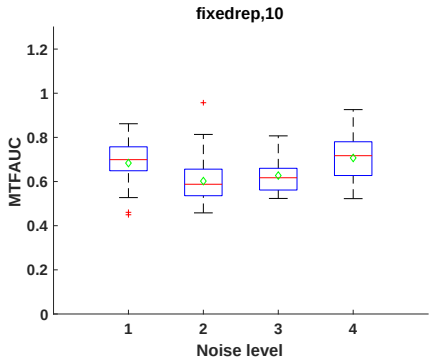
From Table 4.14 and Table 4.15, based on the additive model, the No-Replication



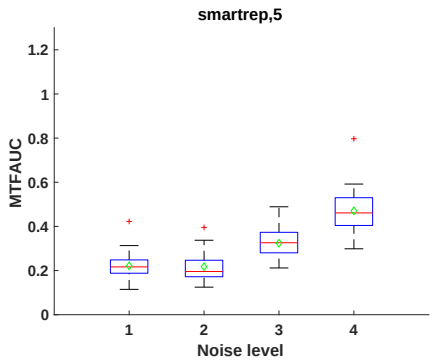
(a)



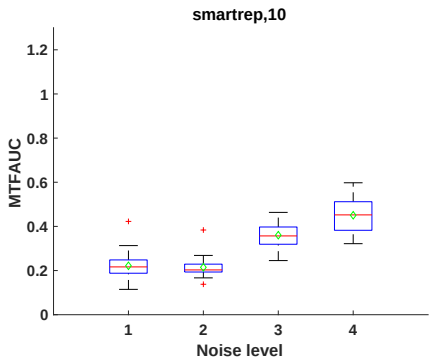
(b)



(c)



(d)



(e)

Figure 4.35: MTFAUC box-plots of Full-Factorial design results for Levy function

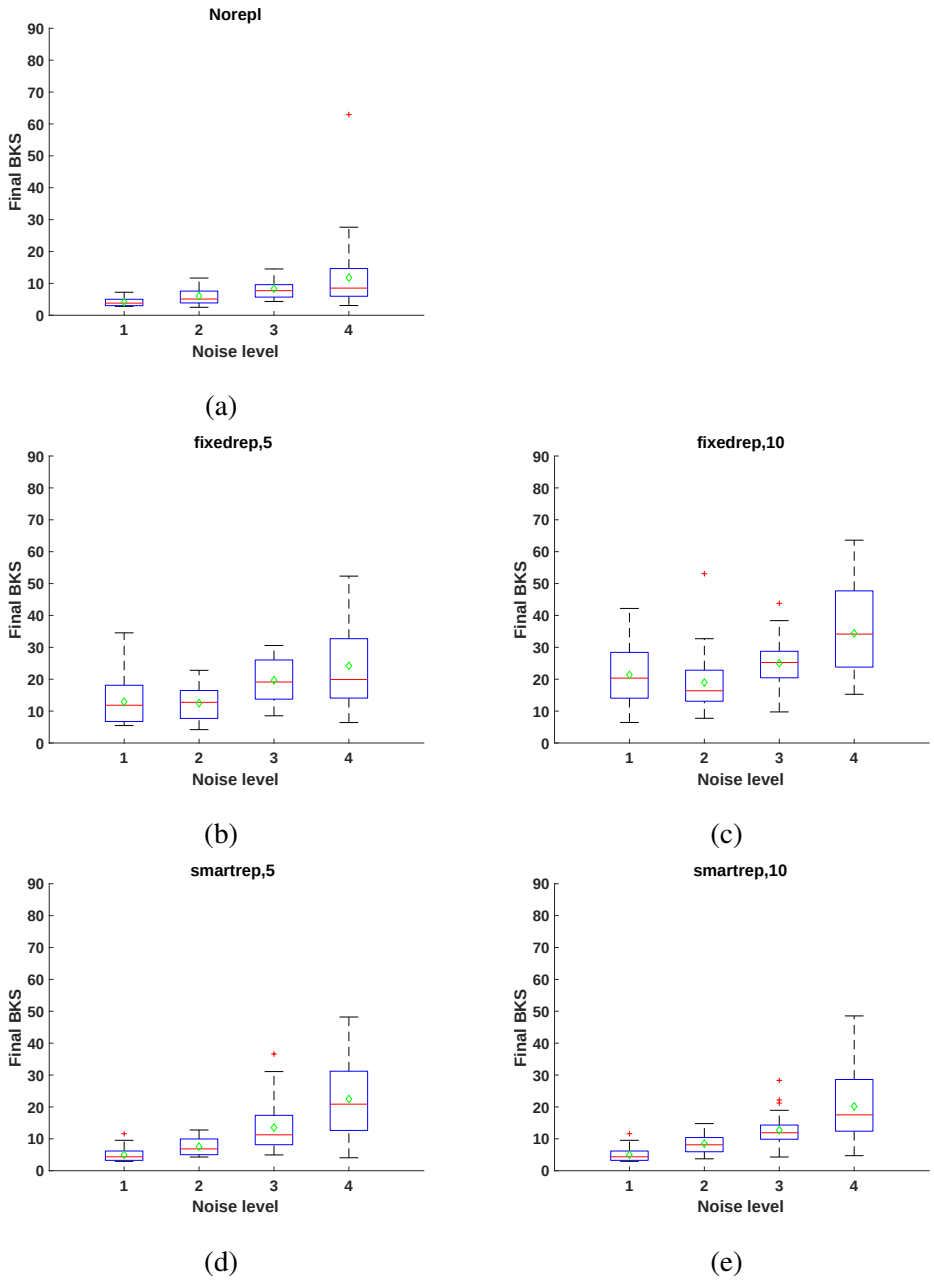


Figure 4.36: Final BSMS box-plots of Full-Factorial design results for Levy function

approach (reference population) statistically outperforms all the other approaches in MTFAUC. We see the noise effect has not been mitigated and is influential on the response, overall. Note that we have No-Replication approach in this analysis besides the Replication approaches. In the OA design, we analyzed the No-Replication and Replication approaches separately (Table 4.7 and Table 4.9). We observe that the No-Replication is not robust to the noise effect and Replication approaches are. Hence, the reason that the highest level of noise factor is significant in this section might be due to No-Replication's non-robustness in handling the noise effect. From Table 4.15, there is a significant $noise{*}repType$ interaction effect. This shows that the effect of noise on the response differs depending on the method applied ($norepl$, $fixedrep, 5$, $fixedrep, 10$, $smartrep, 5$, $smartrep, 10$).

Table 4.14: ANOVA additive model on full-factorial design with TK-MARS

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	0.24003	0.03582	6.701	1.76E-08	***
Rastrigin	0.49064	0.02774	17.684	< 2.00E-16	***
Levy	-0.03802	0.02774	-1.37	0.1767	
noise level=5%	-0.01862	0.03204	-0.581	0.5637	
noise level=10%	0.03634	0.03204	1.134	0.262	
noise level=25%	0.18894	0.03204	5.898	3.16E-07	***
fixedrep,5	0.17057	0.03582	4.762	1.68E-05	***
fixedrep,10	0.30211	0.03582	8.435	3.56E-11	***
smartrep,5	0.0854	0.03582	2.384	0.0209	*
smartep,10	0.08592	0.03582	2.399	0.0202	*
Signif. Codes:	0 '***'	0.001 '**'	0.01 '*'	0.05 '.'	0.1 ' ' 1

Table 4.15: ANOVA full model on full-factorial design with TK-MARS

	Estimate	Std. Error	t value	Pr(>|t|)		
(Intercept)	0.22553	0.055318	4.077	0.000225	***	
Rastrigin	0.490643	0.028889	16.984	< 2.00E-16	***	
Levy	-0.038023	0.028889	-1.316	0.195997		
noise level=5%	-0.007883	0.07459	-0.106	0.91639		
noise level=10%	0.048207	0.07459	0.646	0.521973		
noise level=25%	0.224359	0.07459	3.008	0.004648	**	
fixedrep,5	0.210508	0.07459	2.822	0.007547	**	
fixedrep,10	0.377491	0.07459	5.061	1.10E-05	***	
smartrep,5	0.064265	0.07459	0.862	0.394328		
smartep,10	0.064265	0.07459	0.862	0.394328		
noise=0.05:fixedrep,5	-0.030229	0.105486	-0.287	0.775998		
noise=0.1:fixedrep,5	-0.05072	0.105486	-0.481	0.633399		
noise=0.25:fixedrep,5	-0.078798	0.105486	-0.747	0.459663		
noise=0.05:fixedrep,10	-0.032925	0.105486	-0.312	0.756651		
noise=0.1:fixedrep,10	-0.085798	0.105486	-0.813	0.421079		
noise=0.25:fixedrep,10	-0.182784	0.105486	-1.733	0.091245	.	
noise=0.05:smartrep,5	-0.003959	0.105486	-0.038	0.970256		
noise=0.1:smartrep,5	0.033177	0.105486	0.315	0.75485		
noise=0.25:smartrep,5	0.055323	0.105486	0.524	0.603005		
noise=0.05:smartrep,10	0.01343	0.105486	0.127	0.89936		
noise=0.1:smartrep,10	0.044018	0.105486	0.417	0.678819		
noise=0.25:smartrep,10	0.029177	0.105486	0.277	0.783592		
Signif. Codes:	0 '***'	0.001 '**'	0.01 '*'	0.05 '.'	0.1 ' '	1

- **RBF: Interpolating Model**

Figures 4.37, 4.40 and 4.43 show the performance of the algorithm using different approaches across different noise levels for different test functions. In these plots and the corresponding ANOVA, Tables 4.16-4.17, the surrogate model applied is RBF.

The plots in Figure 4.37 confirm that the deterministic situation, No-Replication, is the best option. However, as the noise level increases, $smartrep, 10$ outperforms $norepl$. We can see that $smartrep, 10$ is more competitive when we use the interpolating model, RBF for the Rosenbrock function. Furthermore, looking closer into the plots, one can observe that under the highest uncertainty level

fixedrep, 10's performance becomes close to *smartrep*, 10, but, overall, Smart-Replication outperforms the Fixed-Replication approach.

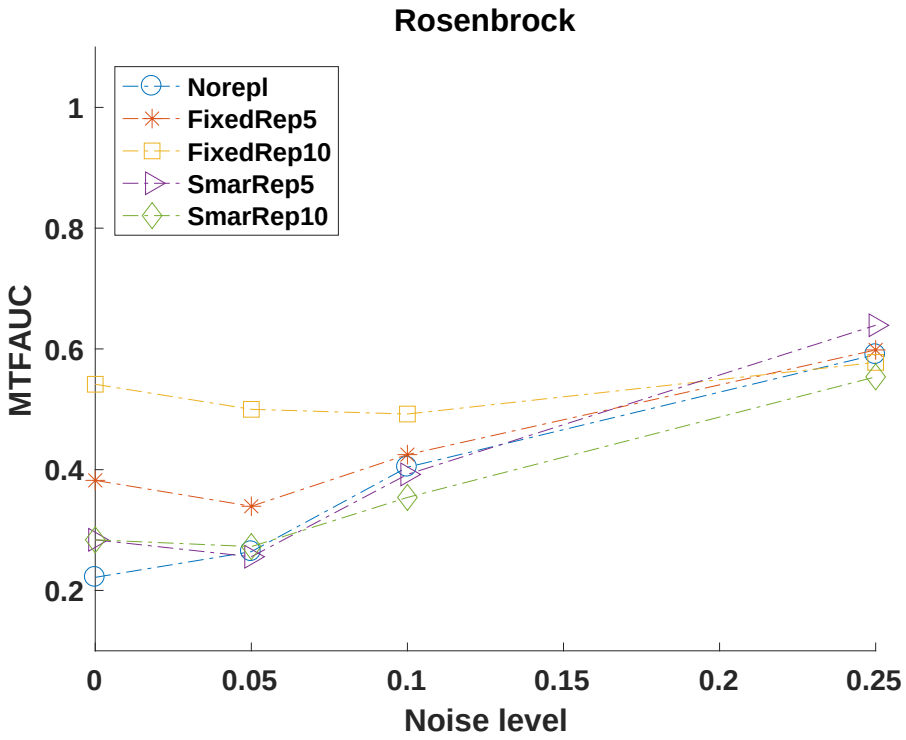


Figure 4.37: MTFAUC of different strategies with RBF across different noise levels

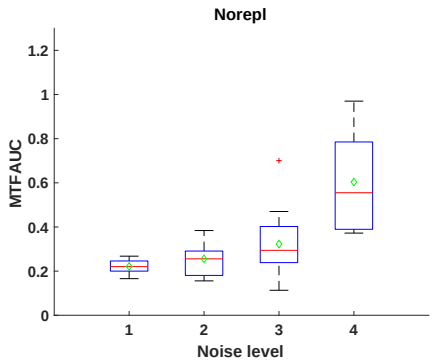
From Figure 4.38, we can see the box-plots of the MTFAUC values for 30 different runs on different noise levels with different methods, Figures 4.38a-4.38e. One can verify that *smartrep*, 10 is the most robust approach to the uncertainty level and randomness since it has the shortest box as the noise level increases. *smartrep*, 10 has the lowest MTFAUC on average, as well. *fixedrep*, 10 is competitive under a higher level of uncertainty and is even more robust to different

noise levels. This makes sense, as replicating cancels out noise effects if it does not exceed the maximum number of function evaluations. Clearly, $norepl$ is the best option in the no-noise case since replication wastes function evaluations when there is no noise associated with the black-box function.

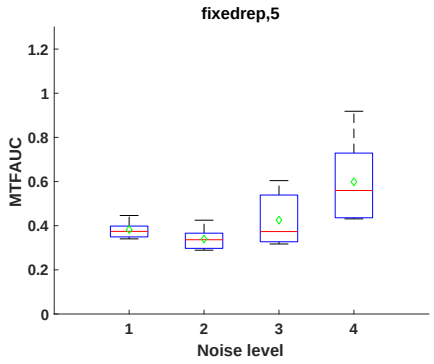
From Figures 4.39, observe that the variance of BSMS after 1000 function evaluations for 30 different runs at each considered noise level. As is observed, $smartrep, 10$ comparatively outperforms in terms of the BSMS at the end. $fixedrep, 10$ and $smartrep, 10$ are robust to different noise levels in terms of finding the BSMS.

Looking into the Rastrigin function, Figure 4.40, we can see that there is a small difference between different methods, especially at the highest level of uncertainty. Because of the highly fluctuating behavior of the Rastrigin function with several local optima, we cannot make a conclusion from the experiment. No-replication slightly outperforms all the other methods, since exploration overcomes the replication in the case of having a complicated function like Rastrigin. It can be seen that Replication approaches with 5 replications are slightly competitive with No-Replication at a very high level of noise. In addition, as the noise level increases, an optimum is harder to obtain.

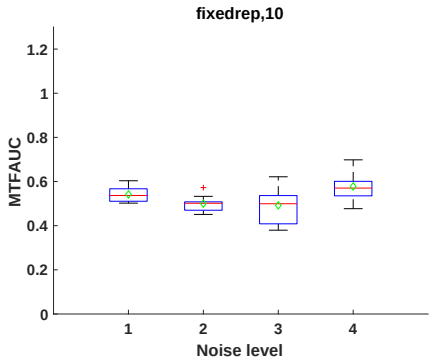
Figure 4.41, shows the box-plot of the MTFAUC values for 30 different runs on different noise levels with different methods, Figures 4.41a-4.41e. For the Rastrigin function, which is a highly fluctuating function, note that $fixedrep, 10$, has the shortest box and is the most robust approach to randomness and also to different noise levels; no difference in the means across noise levels is observed. However, in lower noise levels, $fixeedrep, 10$ has slightly higher average MTFAUC.



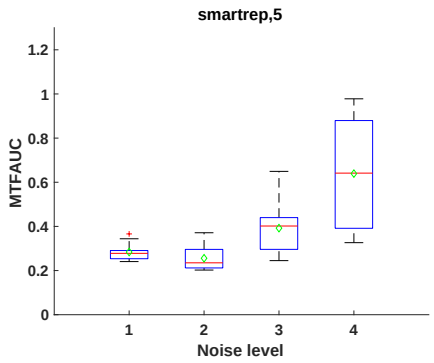
(a)



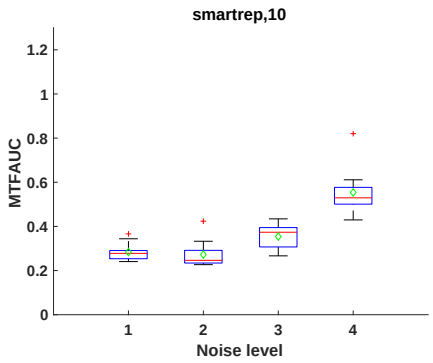
(b)



(c)



(d)



(e)

Figure 4.38: MTFAUC box-plots of Full-Factorial design results for Rosenbrock function with RBF

htbp

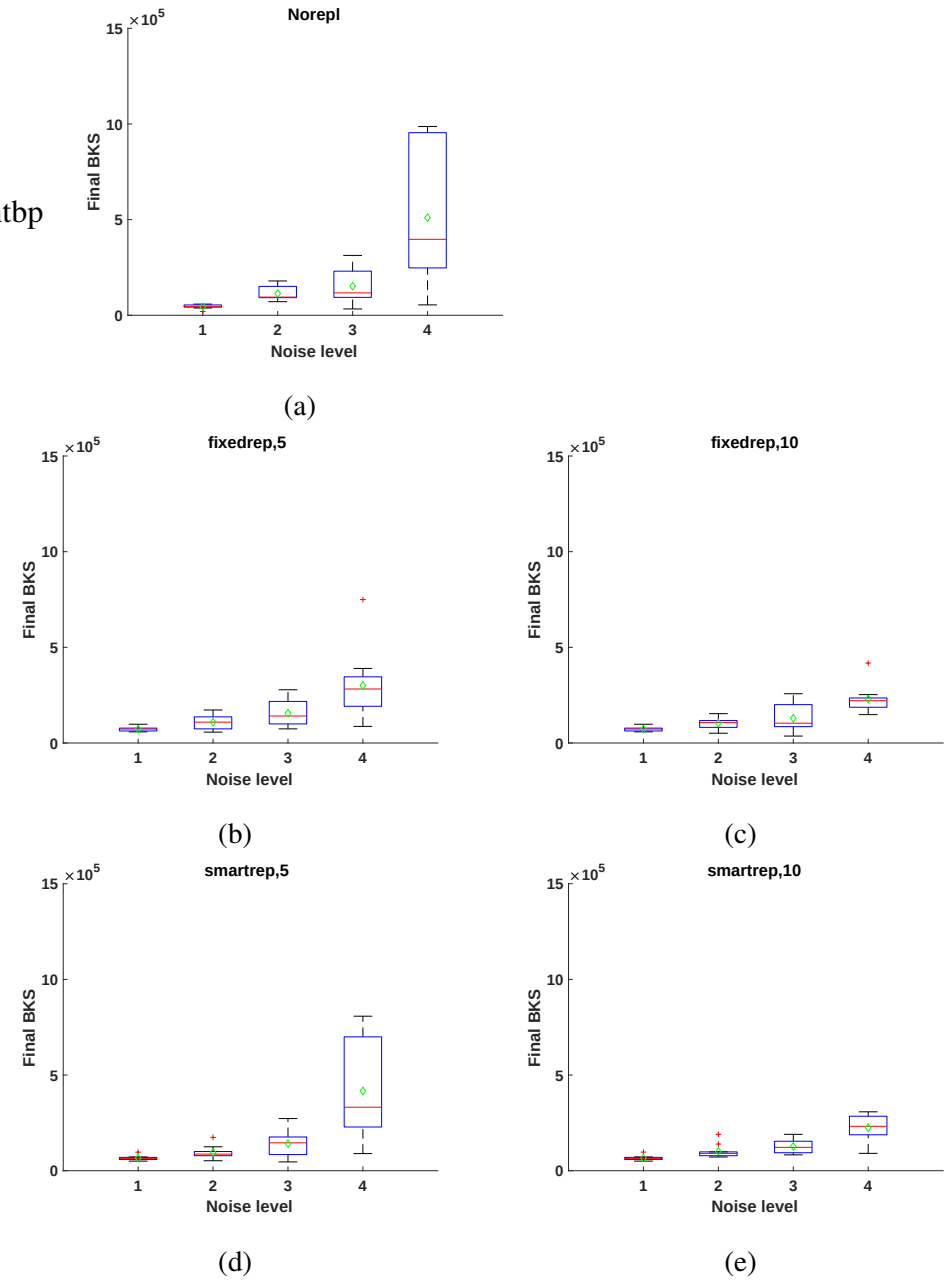


Figure 4.39: Final BSMS box-plots of Full-Factorial design results for Rosenbrock function

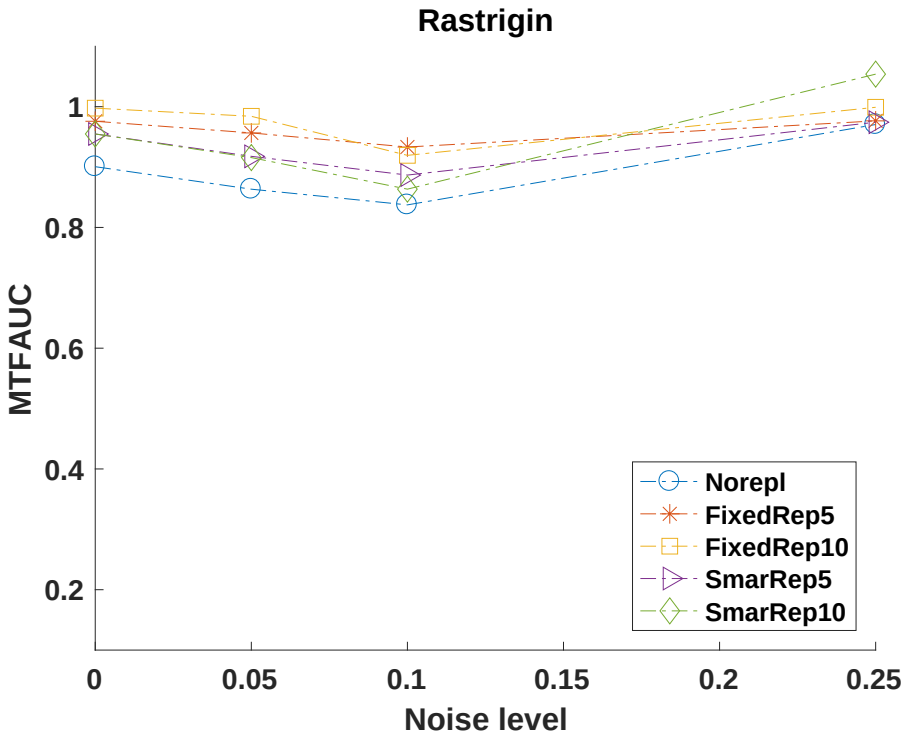


Figure 4.40: MTFAUC of different strategies with RBF across different noise levels

This shows that replication helps to mitigate the fluctuations and uncertainties associated with the black-box function.

Figures 4.42 shows the variance of the BSMS after 1000 function evaluations for 30 different runs at each considered noise level. Note that $frixedrep, 10$ has the shortest box comparatively; however, it has larger BSMS, comparatively. $norepl$ finds better BSMS at the end of 1000 function evaluations when the uncertainty level is low, as we expected. In this case, $fixedrep, 5$ finds better BSMS at the highest noise level.

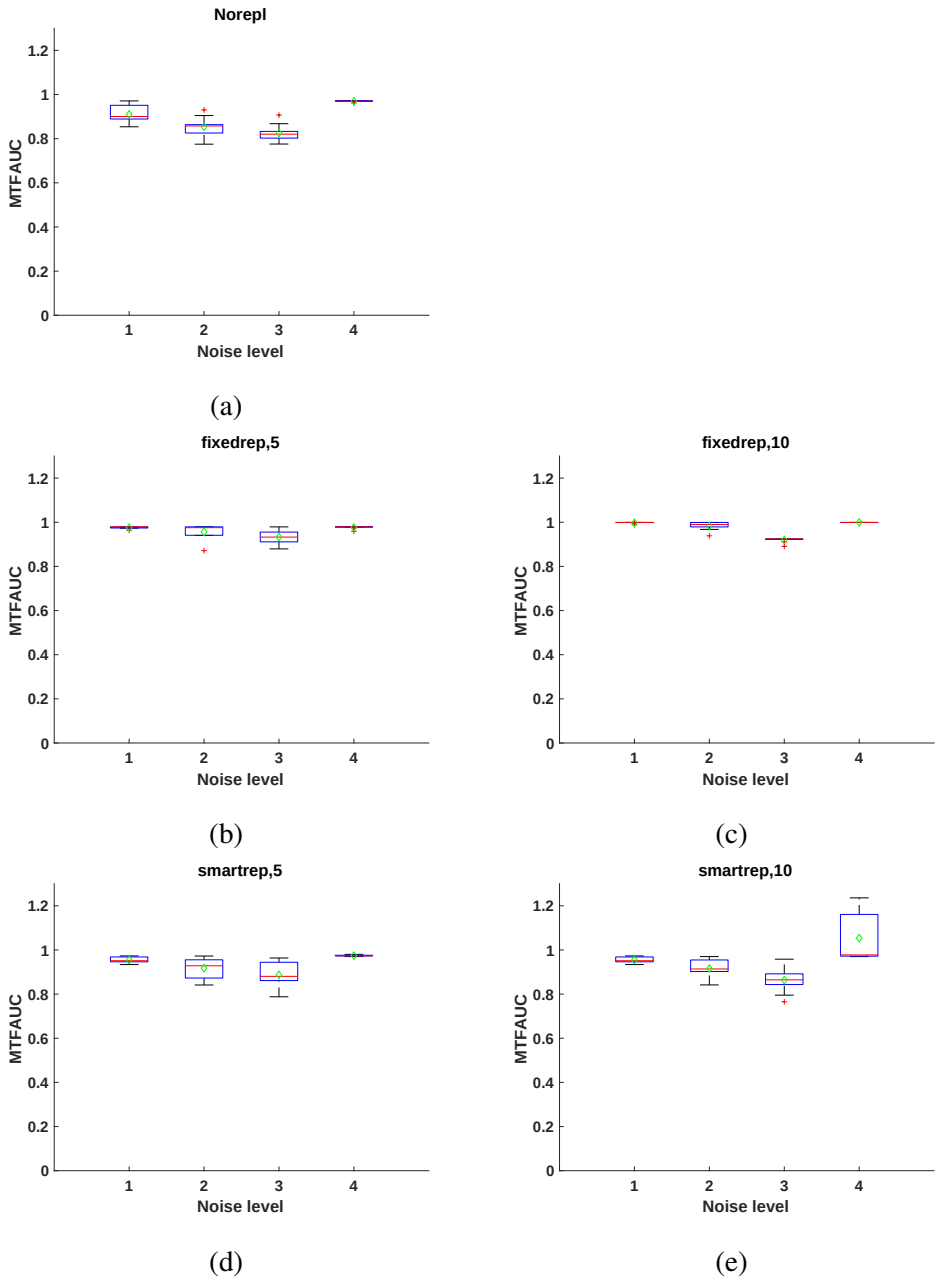


Figure 4.41: MTFAUC box-plots of Full-Factorial design results for Rastrigin function

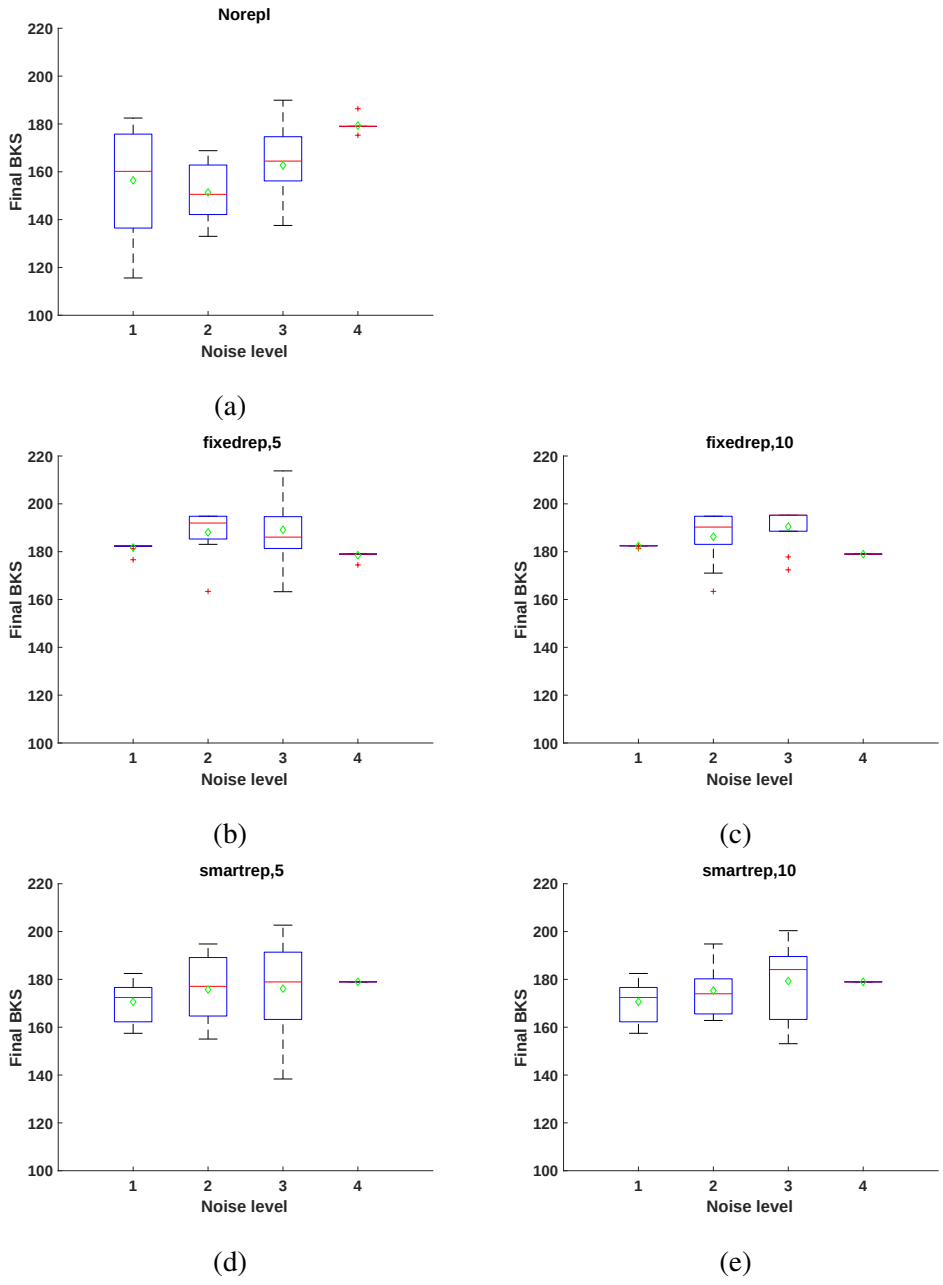


Figure 4.42: Final BSMS box-plots of Full-Factorial design results for Rastrigin function

Looking at Figure 4.43, note that $smartrep, 5$ outperforms other methods under the highest level of uncertainty. $fixedrep, 5$ is competitive, as well. Being a little bit smart helps the optimization process for Levy as the noise level increases.

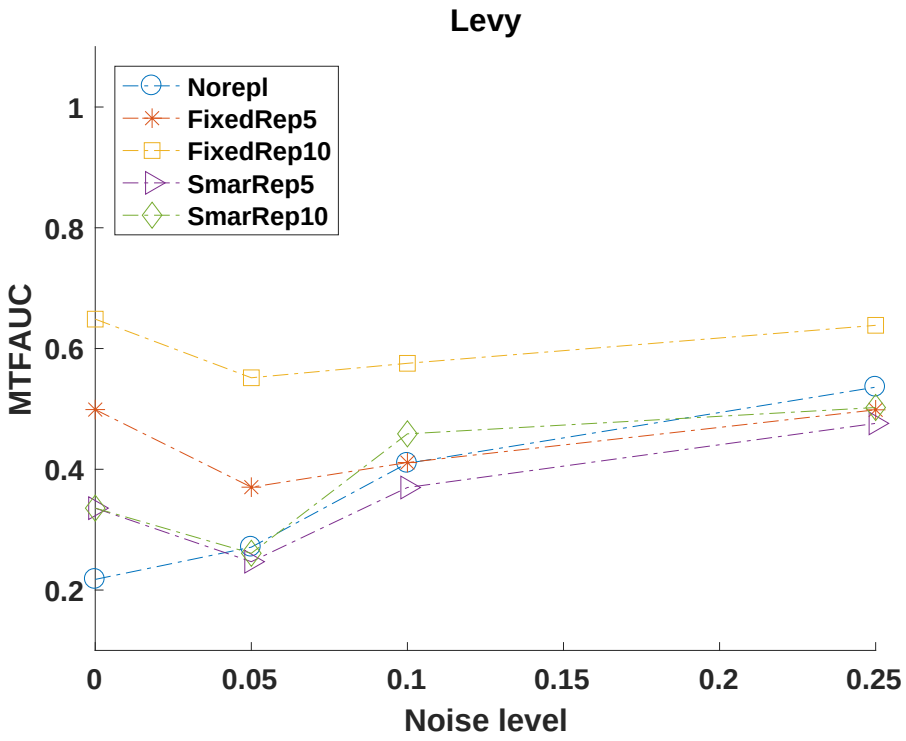


Figure 4.43: MTFAUC of different strategies with RBF across different noise levels

Figure 4.44, shows the box-plot of the MTFAUC values for 30 different runs on different noise levels with different methods, Figures 4.44a-4.44e. One can clearly verify that $smartrep, 5$ has the shortest box overall, except for the no-noise case, which No-replication has the lower MTFAUC on average. However,

in terms of robustness to randomness and the different level of noise, $smartrep, 5$ and *fixedrep*$, 10$ are competitive. However, the MTFAUC is lower for $smartrep, 5$.

Figures 4.45 show the variance of the BSMS after 1000 function evaluations for 30 different runs at each considered noise level. We can see that *fixedrep*$, 10$ is the most robust approach to randomness across different noise levels. However, $smartrep, 5$ has higher quality BSMS.

From the ANOVA on Full Factorial design observations with RBF model (Table 4.16), the No-Replication approach (reference population) statistically outperforms Fixed-Replication. However, Smart-Replication is not significantly different from the No-Replication approach, from the MTFAUC standpoint.

We see that the noise effect has not been mitigated and is influential in the response. This means that for RBF, the algorithm is not robust across different noise levels; we see that a significant interaction between noise level and the approaches. Recall that we have the No-Replication approach in this analysis besides the Replication approaches. In the OA design, we analyze No-Replication and Replication approaches, separately in Tables 4.7 and 4.9. We observe that the No-Replication is not robust across noise levels, but Replication approaches are. Hence, the reason that the highest level of noise factor is significant in this section might be due to No-Replication's non-robustness to the noise effect. From Table 4.17, there is a significant $noise * repType$ interaction. This shows the effect of noise on the response differs depending on the method applied ($norepl$, *fixedrep*$, 5$, *fixedrep*$, 10$, $smartrep, 5$, $smartrep, 10$); this is consistent with our finding in the plots explained above.

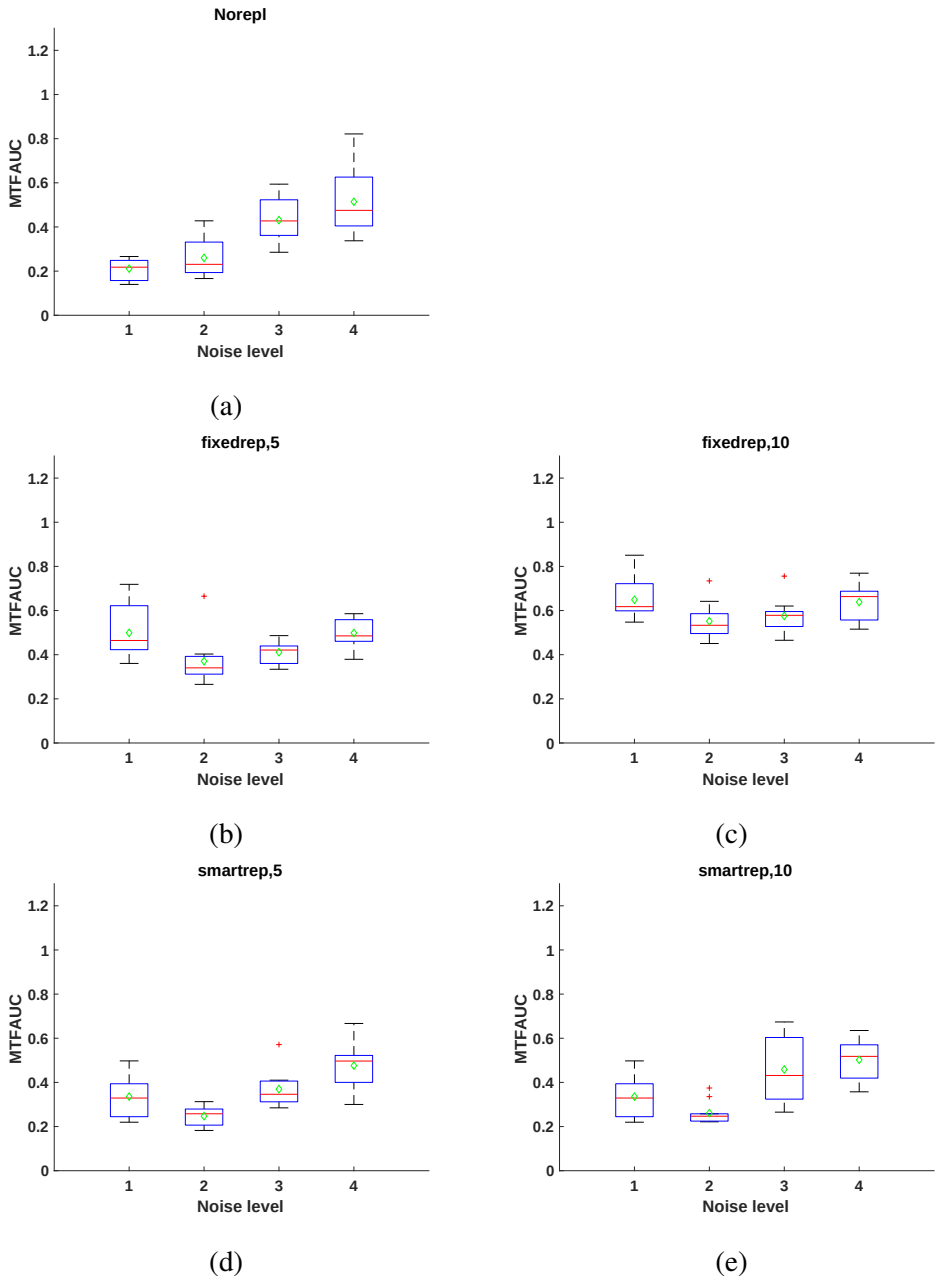


Figure 4.44: MTFAUC box-plots of Full-Factorial design results for Levy function

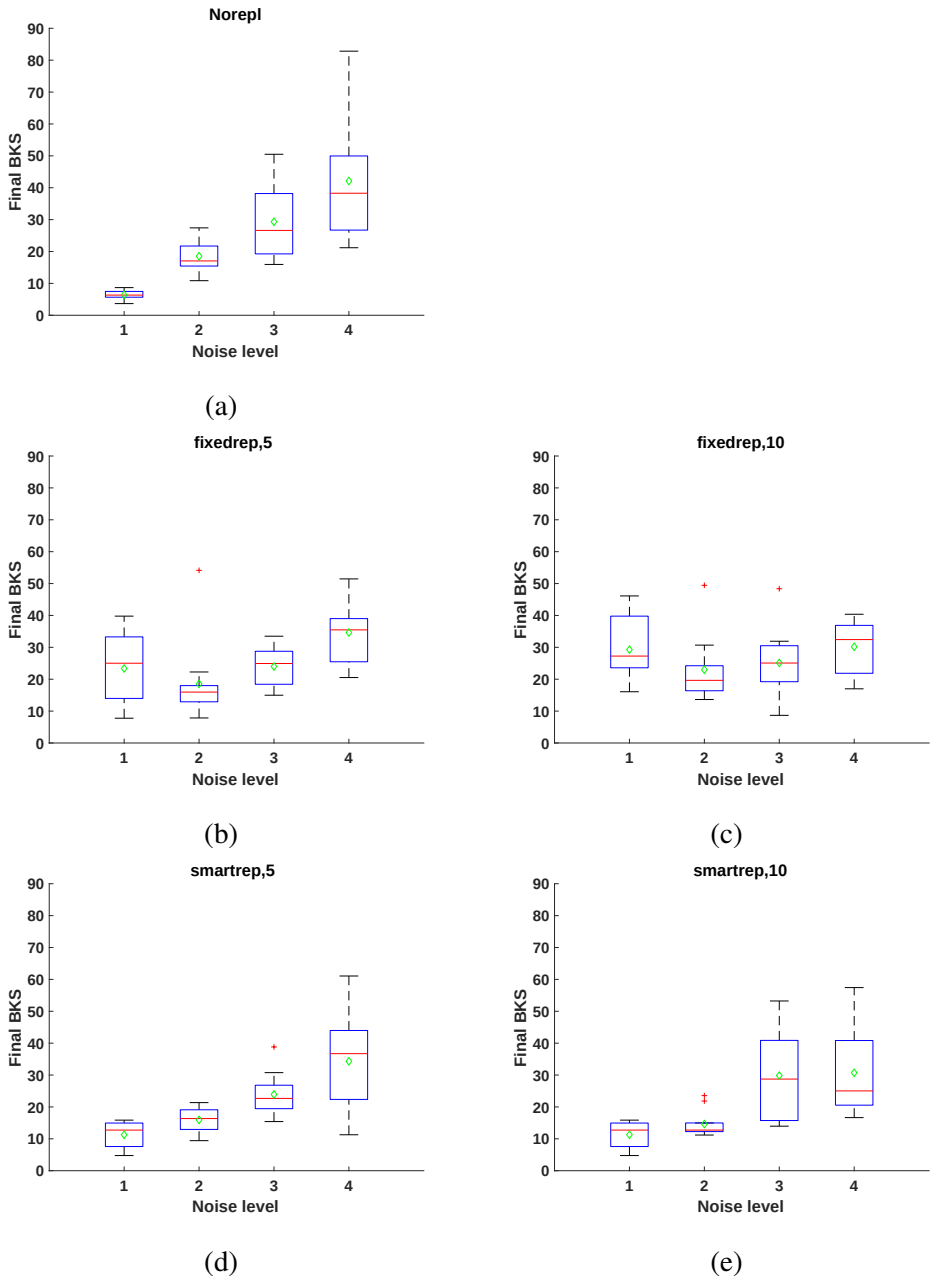


Figure 4.45: Final BSMS box-plots of Full-Factorial design results for Levy function

Table 4.16: ANOVA additive model on full-factorial design with RBF

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	0.33403	0.02905	11.498	1.19E-15	***
Rastrigin	0.52317	0.0225	23.249	<2.00E-16	***
Levy	0.01209	0.0225	0.537	0.5933	
noise level=5%	-0.03754	0.02598	-1.445	0.1548	
noise level=10%	0.01329	0.02598	0.511	0.6114	
noise level=25%	0.13677	0.02598	5.264	2.97E-06	***
fixedrep,5	0.07325	0.02905	2.521	0.0149	*
fixedrep,10	0.16155	0.02905	5.561	1.05E-06	***
smartrep,5	0.0205	0.02905	0.706	0.4836	
smartep,10	0.02688	0.02905	0.925	0.3593	
Signif. Codes:	0 '***'	0.001 '**'	0.01 '*'	0.05 '.'	0.1 ' ' 1

Table 4.17: ANOVA full model on full-factorial design with RBF

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	0.26822	0.04215	6.364	1.8E-07	***
Rastrigin	0.52317	0.02201	23.769	< 2.00E-16	***
Levy	0.01209	0.02201	0.55	0.58587	
noise level=5%	0.01956	0.05683	0.344	0.73261	
noise level=10%	0.10382	0.05683	1.827	0.07557	.
noise level=25%	0.25237	0.05683	4.441	7.5E-05	***
fixedrep,5	0.17237	0.05683	3.033	0.00435	**
fixedrep,10	0.2826	0.05683	4.973	1.45E-05	***
smartrep,5	0.07812	0.05683	1.375	0.17729	
smartep,10	0.07812	0.05683	1.375	0.17729	
noise=0.05:fixedrep,5	-0.08312	0.08037	-1.034	0.30756	
noise=0.1:fixedrep,5	-0.13308	0.08037	-1.656	0.10599	
noise=0.25:fixedrep,5	-0.18028	0.08037	-2.243	0.0308	*
noise=0.05:fixedrep,10	-0.07026	0.08037	-0.874	0.3875	
noise=0.1:fixedrep,10	-0.17072	0.08037	-2.124	0.04023	*
noise=0.25:fixedrep,10	-0.24324	0.08037	-3.026	0.00442	**
noise=0.05:smartrep,5	-0.07082	0.08037	-0.881	0.38374	
noise=0.1:smartrep,5	-0.07902	0.08037	-0.983	0.33174	
noise=0.25:smartrep,5	-0.08063	0.08037	-1.003	0.32209	
noise=0.05:smartrep,10	-0.06127	0.08037	-0.762	0.45055	
noise=0.1:smartrep,10	-0.06988	0.08037	-0.87	0.39001	
noise=0.25:smartrep,10	-0.07382	0.08037	-0.918	0.36417	
Signif. Codes:	0 '***'	0.001 '**'	0.01 '*'	0.05 '.'	0.1 ' ' 1

Now, it is worth discussing Smart-Replication and the maximum number of replications parameter (r_{max}) using TK-MARS and RBF. Smart-Replication fo-

cuses on replicating not all the candidate points but a subset of interesting points, which are probabilistically closer to the BSMS. Since TK-MARS has a better image of the underlying function behavior, it suggests more interesting points for the Smart-Replication rule. Looking into the replication number that each point has had after 1000 function evaluations, there are more points that hit the r_{max} in Smart-Replication using TK-MARS than RBF. TK-MARS makes fewer mistakes in choosing promising candidate points, unlike RBF, which passes through all the points and does not have a good view of the underlying function behavior. RBF then chooses some candidates that might not be interesting after some replications and wastes function evaluations. Consequently, we do not see that many points hitting the r_{max} in Smart-Replication with RBF.

4.6 Conclusion

Regarding the analysis that is presented in this section, TK-MARS outperforms RBF, which is the leading surrogate optimization method in the literature. TK-MARS demonstrates great improvement over RBF in handling unimportant inputs.

Further, we can conclude that using TK-MARS, No-Replication approach outperforms replication approaches in terms of average MTFAUC, which means the area under the curve is smaller. However, Smart-Replication is competitive with No-Replication in terms of robustness to randomness and to different noise levels. For RBF, more replications help robustness in uncertain situations, and Smart-

Replication approach outperforms No-Replication.

TK-MARS is a regression-based model, which does not pass through all the points and fits a surface that minimizes the error based on the distance of the actual values to the fitted line. These models consider the error to predict the future points more accurately. As a result, regression-based models help to have a better approximation for noisy data sets. On the other hand, the goal of replication is to minimize the error for an uncertain point. The gain obtained from replication does not improve the regression-based approximation, since, it has the same effect of replication by passing among the points. As a result, an algorithm that uses TK-MARS, a regression-based model, needs to explore more points rather than replicating fewer points and reduce their error (exploration versus exploitation).

By contrast, interpolating models pass through all the points and capture the noise, as well. For noisy functions, interpolating may be misleading. In this case, replication is very constructive to minimize the error and make the surrogate model more intelligent by giving more information about the underlying true function by averaging over replicated noisy data. We observe that replications improve the performance of an algorithm when we apply RBF in comparison to TK-MARS, which mitigates the noise effect itself and needs more exploration instead.

Chapter 5

Future Work

In this section, we describe the future directions of this research. In Chapter 1 we assumed that the black-box functions considered in this research are box-constrained; however, there might be other constraints, which are black-box, as well. Constrained black-box optimization approaches have been studied in the literature [39,41]. One direction for future study of the current research is to perform the TK-MARS based surrogate optimization approach on constrained black-box optimization problems and measure the performance of the proposed algorithm.

Further, we considered a single objective optimization problem. Employing the TK-MARS based approach and adapt it for the multi-objective black-box optimization problems is worthwhile in order to study the capability of the proposed method for different optimization purposes.

In this research, we focused on the continuous space, and show the results in Chapter 4. To consider categorical variables as well as numerical variables

and investigate the performance of the proposed approach is very functional for the real-world problems, which mostly contain categorical variables. We believe TK-MARS is very capable of handling a mixed categorical/numerical space.

Finally, developing dynamic stopping criteria based on a convergence technique would save function evaluations by automatically stopping as soon the algorithm found a high-quality solution.

Appendices

Table 1: OA Design for No-Replication Study

Fraction of Significant Variables	Noise Level	Test Function	Dimesnison	Pool Size	DOE	EEPA-Distance	EEPA-Number of Candidates	Model
1	0	2	10	22	2	1	2	2
1	0	1	20	21	1	2	5	1
1	0.1	1	20	42	2	2	5	1
1	0.1	2	10	11	1	1	2	2
1	0.25	1	30	62	1	1	5	2
1	0.25	3	10	11	2	2	2	1
0.75	0	3	10	22	1	2	2	2
0.75	0	1	30	31	2	1	5	1
0.75	0.1	1	10	11	1	1	2	1
0.75	0.1	3	30	62	2	2	5	2
0.75	0.25	3	30	31	2	2	2	2
0.75	0.25	1	10	22	1	1	5	1
0.5	0	1	20	21	1	1	5	2
0.5	0	2	10	22	2	2	2	1
0.5	0.1	2	10	22	1	2	5	1
0.5	0.1	1	20	21	2	1	2	2
0.5	0.25	1	20	21	2	1	5	1
0.5	0.25	2	10	22	1	2	2	2
0.25	0	2	10	11	1	2	2	1
0.25	0	1	20	42	2	1	5	2
0.25	0.1	1	10	22	2	1	2	1
0.25	0.1	3	30	31	1	2	5	2
0.25	0.25	3	30	31	2	2	5	2
0.25	0.25	1	10	22	1	1	2	1
1	0	3	20	42	2	1	2	2
1	0	2	30	31	1	2	5	1
1	0.1	2	30	62	2	2	5	1
1	0.1	3	20	21	1	1	2	2
1	0.25	2	10	22	1	1	5	2
1	0.25	1	20	21	2	2	2	1
0.75	0	1	20	42	1	2	2	2
0.75	0	2	10	11	2	1	5	1
0.75	0.1	2	20	21	1	1	2	1
0.75	0.1	1	10	22	2	2	5	2
0.75	0.25	1	10	11	2	2	2	2
0.75	0.25	2	20	42	1	1	5	1
0.5	0	2	30	31	1	1	5	2
0.5	0	3	20	42	2	2	2	1
0.5	0.1	3	20	42	1	2	5	1
0.5	0.1	2	30	31	2	1	2	2
0.5	0.25	2	30	31	2	1	5	1
0.5	0.25	3	20	42	1	2	2	2
0.25	0	3	20	21	1	2	2	1
0.25	0	2	30	62	2	1	5	2
0.25	0.1	2	20	42	2	1	2	1
0.25	0.1	1	10	11	1	2	5	2
0.25	0.25	1	10	11	2	2	5	2
0.25	0.25	2	20	42	1	1	2	1
1	0	1	30	62	2	1	2	2
1	0	3	10	11	1	2	5	1
1	0.1	3	10	22	2	2	5	1
1	0.1	1	30	31	1	1	2	2
1	0.25	3	20	42	1	1	5	2
1	0.25	2	30	31	2	2	2	1
0.75	0	2	30	62	1	2	2	2
0.75	0	3	20	21	2	1	5	1
0.75	0.1	3	30	31	1	1	2	1
0.75	0.1	2	20	42	2	2	5	2
0.75	0.25	2	20	21	2	2	2	2
0.75	0.25	3	30	62	1	1	5	1
0.5	0	3	10	11	1	1	5	2
0.5	0	1	30	62	2	2	2	1
0.5	0.1	1	30	62	1	2	5	1
0.5	0.1	3	10	11	2	1	2	2
0.5	0.25	3	10	11	2	1	5	1
0.5	0.25	1	30	62	1	2	2	2
0.25	0	1	30	31	1	2	2	1
0.25	0	3	10	22	2	1	5	2
0.25	0.1	3	30	62	2	1	2	1
0.25	0.1	2	20	21	1	2	5	2
0.25	0.25	2	20	21	2	2	5	2
0.25	0.25	3	30	62	1	1	2	1
1	0	1	30	31	1	1	1	1
1	0.05	1	30	31	1	1	1	1
1	0.1	1	30	31	1	1	1	1
1	0.25	1	30	31	1	1	1	1

Table 2: OA Design for With-Replication Study

Fraction of Significant Variables	Noise Level	Test Function	Dimension	Model	Number of Replication	Pool Size	DOE	EEPA-distance	EEPA-Number of candidates	Replication Type
1	0.05	3	10	2	4	22	1	2	5	1
1	0.05	1	30	2	2	31	2	1	2	2
1	0.1	3	10	3	2	22	2	2	5	2
1	0.1	1	30	3	4	31	1	1	2	1
1	0.25	1	10	2	2	22	2	1	2	2
1	0.25	2	20	2	9	21	1	2	5	1
0.75	0.05	1	10	3	9	22	1	1	5	1
0.75	0.05	2	20	3	2	21	2	2	2	2
0.75	0.1	1	10	3	2	11	1	1	2	1
0.75	0.1	3	20	3	9	42	2	2	5	2
0.75	0.25	1	10	1	9	11	2	2	2	1
0.75	0.25	3	20	1	2	42	1	1	5	2
0.5	0.05	2	10	3	2	11	2	1	5	2
0.5	0.05	1	30	3	4	62	1	2	2	1
0.5	0.1	2	10	1	4	22	1	1	2	2
0.5	0.1	1	30	1	2	31	2	2	5	1
0.5	0.25	1	20	1	2	21	1	2	5	2
0.5	0.25	3	10	1	4	22	2	1	2	1
0.25	0.05	1	20	2	4	21	2	1	5	1
0.25	0.05	3	10	2	2	22	1	2	2	2
0.25	0.1	2	30	1	2	62	2	2	2	1
0.25	0.1	1	10	1	9	11	1	1	5	2
0.25	0.25	2	30	2	9	31	1	2	2	2
0.25	0.25	1	10	2	2	22	2	1	5	1
1	0.05	1	20	3	9	42	1	2	5	1
1	0.05	2	10	3	4	11	2	1	2	2
1	0.1	1	20	1	4	42	2	2	5	2
1	0.1	2	10	1	9	11	1	1	2	1
1	0.25	2	20	3	4	42	2	1	2	2
1	0.25	3	30	3	2	31	1	2	5	1
0.75	0.05	2	20	1	2	42	1	1	5	1
0.75	0.05	3	30	1	4	31	2	2	2	2
0.75	0.1	2	20	1	4	21	1	1	2	1
0.75	0.1	1	30	1	2	62	2	2	5	2
0.75	0.25	2	20	2	2	21	2	2	2	1
0.75	0.25	1	30	2	4	62	1	1	5	2
0.5	0.05	3	20	1	4	21	2	1	5	2
0.5	0.05	2	10	1	9	22	1	2	2	1
0.5	0.1	3	20	2	9	42	1	1	2	2
0.5	0.1	2	10	2	4	11	2	2	5	1
0.5	0.25	2	30	2	4	31	1	2	5	2
0.5	0.25	1	20	2	9	42	2	1	2	1
0.25	0.05	2	30	3	9	31	2	1	5	1
0.25	0.05	1	20	3	4	42	1	2	2	2
0.25	0.1	3	10	2	4	22	2	2	2	1
0.25	0.1	2	20	2	2	21	1	1	5	2
0.25	0.25	3	10	3	2	11	1	2	2	2
0.25	0.25	2	20	3	4	42	2	1	5	1
1	0.05	2	30	1	2	62	1	2	5	1
1	0.05	3	20	1	9	21	2	1	2	2
1	0.1	2	30	2	9	62	2	2	5	2
1	0.1	3	20	2	2	21	1	1	2	1
1	0.25	3	30	1	9	62	2	1	2	2
1	0.25	1	10	1	4	11	1	2	5	1
0.75	0.05	3	30	2	4	62	1	1	5	1
0.75	0.05	1	10	2	9	11	2	2	2	2
0.75	0.1	3	30	2	9	31	1	1	2	1
0.75	0.1	2	10	2	4	22	2	2	5	2
0.75	0.25	3	30	3	4	31	2	2	2	1
0.75	0.25	2	10	3	9	22	1	1	5	2
0.5	0.05	1	30	2	9	31	2	1	5	2
0.5	0.05	3	20	2	2	42	1	2	2	1
0.5	0.1	1	30	3	2	62	1	1	2	2
0.5	0.1	3	20	3	9	21	2	2	5	1
0.5	0.25	3	10	3	9	11	1	2	5	2
0.5	0.25	2	30	3	2	62	2	1	2	1
0.25	0.05	3	10	1	2	11	2	1	5	1
0.25	0.05	2	30	1	9	62	1	2	2	2
0.25	0.1	1	20	3	9	42	2	2	2	1
0.25	0.1	3	30	3	4	31	1	1	5	2
0.25	0.25	1	20	1	4	21	1	2	2	2
0.25	0.25	3	30	1	9	62	2	1	5	1
1	0	1	30	1	4	31	1	1	1	2
1	0.05	1	30	1	4	31	1	1	1	2
1	0.1	1	30	1	4	31	1	1	1	2
1	0.25	1	30	1	4	31	1	1	1	2

Bibliography

[1] H. Liu, S. Lee, M. Kim, H. Shi, J. T. Kim, K. L. Wasewar, and C. Yoo, "Multi-objective optimization of indoor air quality control and energy consumption minimization in a subway ventilation system," *Energy and Buildings*, vol. 66, pp. 553–561, 2013.

[2] P. Kung, "Multivariate modeling for a multiple stage, multiple objective green building framework," 2013.

[3] J. April, M. Better, F. Glover, J. Kelly, and M. Laguna, "Enhancing business process management with simulation optimization," in *Proceedings of the 38th conference on Winter simulation*. Winter Simulation Conference, 2006, pp. 642–649.

[4] N. Martinez, H. Anahideh, J. M. Rosenberger, D. Martinez, V. C. Chen, and B. P. Wang, "Global optimization of non-convex piecewise linear regression splines," *Journal of Global Optimization*, vol. 68, no. 3, pp. 563–586, 2017.

[5] V. Picheny, R. Trépos, and P. Casadebaig, "Optimization of black-box models with uncertain climatic inputs—application to sunflower ideotype design," *PloS one*, vol. 12, no. 5, p. e0176815, 2017.

[6] D. G. Krige, "A statistical approach to some mine valuation and allied problems on the witwatersrand: By dg krige," Ph.D. dissertation, University of the Witwatersrand, 1951.

[7] M. J. Powell, "Radial basis funcitionn for multivariable interpolation: A review," in *IMA Conference on Algorithms for the Approximation of Functions ans Data*. RMCS, 1985, pp. 143–167.

[8] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*. CRC press, 1984.

[9] J. H. Friedman, "Multivariate adaptive regression splines," *The annals of statistics*, pp. 1–67, 1991.

[10] M. Papadrakakis, N. D. Lagaros, and Y. Tsompanakis, "Structural optimization using evolution strategies and neural networks," *Computer methods in applied mechanics and engineering*, vol. 156, no. 1-4, pp. 309–333, 1998.

[11] S. M. Clarke, J. H. Griebsch, and T. W. Simpson, "Analysis of support vector regression for approximation of complex engineering analyses," *Journal of mechanical design*, vol. 127, no. 6, pp. 1077–1087, 2005.

[12] V. Pilla, "Robust airline fleet assignment," 2007.

[13] V. Picheny, D. Ginsbourger, Y. Richet, and G. Caplin, "Quantile-based optimization of noisy computer experiments with tunable precision," *Technometrics*, vol. 55, no. 1, pp. 2–13, 2013.

[14] J. F. Dickson, *An exploration and exploitation pareto approach to surrogate optimization*. The University of Texas at Arlington, 2014.

[15] I. M. Sobol', "On the distribution of points in a cube and the approximate evaluation of integrals," *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, vol. 7, no. 4, pp. 784–802, 1967.

[16] I. M. Sobol, "Uniformly distributed sequences with an additional uniform property," *USSR Computational Mathematics and Mathematical Physics*, vol. 16, no. 5, pp. 236–242, 1976.

[17] A. S. Hedayat, N. J. A. Sloane, and J. Stufken, *Orthogonal arrays: theory and applications*. Springer Science & Business Media, 2012.

[18] J. Neter, M. H. Kutner, C. J. Nachtsheim, and W. Wasserman, *Applied linear statistical models*. Irwin Chicago, 1996, vol. 4.

[19] R. Unal, R. Lepsch, W. Engelund, and D. Stanley, "Approximation model building and multidisciplinary design optimization using response surface methods," in *6th Symposium on Multidisciplinary Analysis and Optimization*, 1996, p. 4044.

[20] T. Simpson, F. Mistree, J. Korte, and T. Mauery, "Comparison of response surface and kriging models for multidisciplinary design optimization," in *7th*

AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, 1998, p. 4755.

[21] A. W. Moore, J. G. Schneider, J. A. Boyan, and M. S. Lee, "Q2: Memory-based active learning for optimizing noisy continuous functions," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 4. IEEE, 2000, pp. 4095–4102.

[22] R. F. Gunst, "Response surface methodology: process and product optimization using designed experiments," 1996.

[23] S. Subrat, "Multivariate adaptive regression spline based framework for statistically parsimonious adaptive dynamic programming," Ph.D. dissertation, Ph. D. dissertation, Dept. Ind. Manuf. Syst. Eng., Univ. Texas Arlington, Arlington, TX, USA, 2011.

[24] D. Martinez, "Variants of multivariate adaptive regression splines (mars): Convex vs. non-convex, piecewise-linear vs. smooth and sequential algorithms," Ph.D. dissertation, Ph. D. thesis, The University of Texas at Arlington, 2013.

[25] Y. Cao, H. Lin, T. Z. Wu, and Y. Yu, "Penalized spline estimation for functional coefficient regression models," *Computational statistics & data analysis*, vol. 54, no. 4, pp. 891–905, 2010.

[26] J. Z. Huang and H. Shen, "Functional coefficient regression models for non-linear time series: A polynomial spline approach," *Scandinavian journal of statistics*, vol. 31, no. 4, pp. 515–534, 2004.

[27] Q. Song and L. Yang, "Oracally efficient spline smoothing of nonlinear additive autoregression models with simultaneous confidence band," *Journal of Multivariate Analysis*, vol. 101, no. 9, pp. 2008–2025, 2010.

[28] S. Miyata and X. Shen, "Free-knot splines and adaptive knot selection," *Journal of the Japan Statistical Society*, vol. 35, no. 2, pp. 303–324, 2005.

[29] E. K. Koc and C. Iyigun, "Restructuring forward step of mars algorithm using a new knot selection procedure based on a mapping approach," *Journal of Global Optimization*, vol. 60, no. 1, pp. 79–102, 2014.

[30] J. C. Tsai and V. C. Chen, "Flexible and robust implementations of multivariate adaptive regression splines within a wastewater treatment stochastic dynamic program," *Quality and Reliability Engineering International*, vol. 21, no. 7, pp. 689–699, 2005.

[31] S. Crino and D. E. Brown, "Global optimization with multivariate adaptive regression splines," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 2, pp. 333–340, 2007.

[32] M. Costas, J. Díaz, L. Romera, and S. Hernández, "A multi-objective surrogate-based optimization of the crashworthiness of a hybrid impact ab-

sorber," *International Journal of Mechanical Sciences*, vol. 88, pp. 46–54, 2014.

[33] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments," *Statistical science*, pp. 409–423, 1989.

[34] A. J. Booker, J. E. Dennis, P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset, "A rigorous framework for optimization of expensive functions by surrogates," *Structural optimization*, vol. 17, no. 1, pp. 1–13, 1999.

[35] H. Rocha, "On the selection of the most adequate radial basis function," *Applied Mathematical Modelling*, vol. 33, no. 3, pp. 1573–1583, 2009.

[36] G. B. Wright, "Radial basis function interpolation: numerical and analytical developments," 2003.

[37] R. G. Regis and C. A. Shoemaker, "Constrained global optimization of expensive black box functions using radial basis functions," *Journal of Global optimization*, vol. 31, no. 1, pp. 153–171, 2005.

[38] H.-M. Gutmann, "A radial basis function method for global optimization," *Journal of global optimization*, vol. 19, no. 3, pp. 201–227, 2001.

[39] R. G. Regis, "Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points," *Engineering Optimization*, vol. 46, no. 2, pp. 218–243, 2014.

[40] R. Regis, "Stochastic radial basis function algorithms for large-scale optimization involving expensive black-box objective and constraint functions," *Computers & Operations Research*, vol. 38, no. 5, pp. 837–853, 2011.

[41] R. G. Regis and C. A. Shoemaker, "A stochastic radial basis function method for the global optimization of expensive functions," *INFORMS Journal on Computing*, vol. 19, no. 4, pp. 497–509, 2007.

[42] R. G. Regis, "Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 326–347, 2014.

[43] R. Datta and R. G. Regis, "A surrogate-assisted evolution strategy for constrained multi-objective optimization," *Expert Systems with Applications*, vol. 57, pp. 270–284, 2016.

[44] H. Dong, B. Song, P. Wang, and S. Huang, "A kind of balance between exploitation and exploration on kriging for global optimization of expensive functions," *Journal of Mechanical Science and Technology*, vol. 29, no. 5, pp. 2121–2133, 2015.

[45] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[46] J. Knowles, "Parego: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 50–66, 2006.

[47] B. Bischl, S. Wessing, N. Bauer, K. Friedrichs, and C. Weihs, "Moi-mbo: Multiobjective infill for parallel model-based optimization," in *International Conference on Learning and Intelligent Optimization*. Springer, 2014, pp. 173–186.

[48] T. Krityakierne, T. Akhtar, and C. A. Shoemaker, "Sop: parallel surrogate global optimization with pareto center selection for computationally expensive single objective problems," *Journal of Global Optimization*, vol. 66, no. 3, pp. 417–437, 2016.

[49] S. Jakobsson, M. Patriksson, J. Rudholm, and A. Wojciechowski, "A method for simulation based optimization using radial basis functions," *Optimization and Engineering*, vol. 11, no. 4, pp. 501–532, 2010.

[50] D. Huang, T. T. Allen, W. I. Notz, and N. Zeng, "Global optimization of stochastic black-box systems via sequential kriging meta-models," *Journal of global optimization*, vol. 34, no. 3, pp. 441–466, 2006.

[51] V. Picheny, T. Wagner, and D. Ginsbourger, "A benchmark of kriging-based infill criteria for noisy optimization," *Structural and Multidisciplinary Optimization*, vol. 48, no. 3, pp. 607–626, 2013.

[52] E. Davis and M. Ierapetritou, "Adaptive optimisation of noisy black-box functions inherent in microscopic models," *Computers & chemical engineering*, vol. 31, no. 5, pp. 466–476, 2007.

[53] J.-B. Grill, M. Valko, and R. Munos, "Black-box optimization of noisy functions with unknown smoothness," in *Advances in Neural Information Processing Systems*, 2015, pp. 667–675.

[54] G. Jekabsons, "Rbf: Radial basis function interpolation for matlab/octave," *Riga Technical University, Latvia. version*, vol. 1, 2009.

[55] S. Surjanovic and D. Bingham, "Virtual library of simulation experiments: Test functions and datasets," Retrieved April 25, 2017, from http://www.sfu.ca/~ssurjano.

[56] N. Hansen, S. Finck, R. Ros, and A. Auger, "Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions," Ph.D. dissertation, INRIA, 2009.

[57] B. Addis and M. Locatelli, "A new class of test functions for global optimization," *Journal of Global Optimization*, vol. 38, no. 3, pp. 479–501, 2007.

[58] B. Balasundaram and S. Butenko, "Constructing test functions for global optimization using continuous formulations of graph problems," *Optimization Methods and Software*, vol. 20, no. 4-5, pp. 439–452, 2005.

[59] M. Jamil and X.-S. Yang, "A literature survey of benchmark functions for global optimisation problems," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 4, no. 2, pp. 150–194, 2013.

[60] R. C. Bose and K. A. Bush, "Orthogonal arrays of strength two and three," *The Annals of Mathematical Statistics*, pp. 508–524, 1952.

[61] R. L. Plackett and J. P. Burman, "The design of optimum multifactorial experiments," *Biometrika*, vol. 33, no. 4, pp. 305–325, 1946.

[62] W. F. Kuhfeld, "Orthogonal arrays library," Retrieved March 21, 2017, from http://support.sas.com/techsup/technote/ts723.html.