VARIANTS OF MULTIVARIATE ADAPTIVE REGRESSION SPLINES (MARS): CONVEX VS. NONCONVEX, PIECEWISE-LINEAR VS. SMOOTH AND SEQUENTIAL ALGORITHMS.

by

DIANA LUISA MARTINEZ CEPEDA

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 2013

To my mother, father, brother and my twin sister

for their unconditional support and infinite love.

ACKNOWLEDGEMENTS

I have met many people during this fascinating experience that have really made a contribution toward making this happen. Not only because of the professional and academic aspects but also for the emotional and motivational support that helped me adapt to a new country. Being apart from my family and from my love ones has truly been a worthwhile sacrifice.

There are certainly a lot of people I want to thank for helping me to achieve this important goal in my life. First of all, I would like to express my deepest gratitude to one of the women from whom I have learned the most and who I truly admire, my supervisor professor Dr. Victoria Chen. It is incredible the amount of knowledge I learned working with her. Her passion was always my motivation. Special thanks also to my committee members, Dr. Jay Rosenberger and Dr. Li Zeng for their valuable support. Thank you as well to Dr. Imrhan for all his advice during my graduate studies.

I want to offer my gratitude to a person that, without whom, this would certainly have never been possible, Dr. Liles, thank you for giving me the opportunity to study at the University of Texas at Arlington. I would also like to thank my friend Dr. Manuel Quevedo for the crucial help I received, for his positive enthusiasm in any circumstance and for teaching me that there is always not only plan A and B, but all the way to Z!

Furthermore I would also appreciate the financial support provided by the Mexican institution CONACYT, the Mexican National Council for Science and Technology. Thank you also to Drew Casani and to Joe Crosswell for allowing me to be part of the TMAC team. Being around very knowledgeable individuals and having the opportunity to collaborate on many different projects have made me grow professionally and increase my confidence level significantly. It has truly been a pleasure working for TMAC.

A special thanks to Dr. Thomas Shih for helping me understand and continue with his research work. Also thank you to Giles DSilva for spending time with me explaining programming languages.

Thank you to all the IMSE (Industrial and Manufacturing Systems Engineering) professors and staff and to all my friends who always built a very nice environment, especially to my COSMOS (Center on Stochastic Modeling, Optimization, & Statistics) friends and Misa Faezipour.

Finally but not least, invaluable thanks to my family who is the most valuable treasure I have in my life. Thanks for trusting me. This is totally for you all! Thank you to my mother Maria Luisa Cepeda Tijerina and father Macedonio Martinez Gonzalez for their constant support and help given during my studies regardless of the distance. Thank you to my brother Macedonio Martinez for his clever advice and especially thank you to my twin soul, my sister Nadia Martinez who was my constant companion. I simply cannot imagine this experience without her. Thank you also to a very special person in my life, Carlos Recio.

Living in the United States for these past years has been absolutely one of the most important experiences in my life not only because of the professional achievements I have earned but also because of the amazing people I have been fortunate to meet. I feel truly blessed, thank you God for all!

July 16, 2013

ABSTRACT

VARIANTS OF MULTIVARIATE ADAPTIVE REGRESSION SPLINES (MARS): CONVEX VS. NONCONVEX, PIECEWISE-LINEAR VS. SMOOTH AND SEQUENTIAL ALGORITHMS.

DIANA LUISA MARTINEZ CEPEDA, Ph.D.

The University of Texas at Arlington, 2013

Supervising Professor: Victoria C. P. Chen

Multivariate adaptive regression splines (MARS) is a statistical modeling method used to represent high-dimensional data with interactions. It uses different algorithms to select the terms to be included in the approximation model that best represent the data. In addition, it performs a variable selection, therefore the most significant predictors are shown in the final model.

Design and analysis of computer experiments (DACE) is a statistical technique for creating approximations (called metamodels) of computer models. The goal of DACE is to efficiently predict the response value of a computer model. MARS has been used as a metamodel in DACE techniques. One of the most important properties of MARS is that, it is flexible in its structure, but it can still be restricted to satisfy certain characteristics. For optimization problems in which there is an unknown function that must be approximated, DACE approach could be applied. In stochastic dynamic programming (SDP) for example, MARS could be used to approximate the unknown future value function. However if the function to optimize is known to follow certain properties such as convexity, then the statistical metamodel, in this case MARS should seek to satisfy these properties.

Based on the original MARS structure, different variants have been developed including the ability to model a convex function or a piecewise-linear function, or a smoothing option using a quintic routine. By enabling these variants, MARS modeling facilitates the optimization process.

DACE has had an enormous contribution for studying complex systems, however one consistent concern for researchers is computational time. As researchers seek to study more and more complex systems, corresponding computer models continue to push the limits of computing power. To overcome this drawback and to reduce computational effort, efficient sequential approaches have been studied.

This research focuses its efforts on the development of sequential approaches based on the MARS model. The objective is to sequentially update the approximation function using current and new input data points. Additionally, by using fewer input data points, an accurate prediction of the unknown function could be obtained faster. This could also facilitate the computational effort of an optimization process. Different case studies are shown in order to test the different MARS variants and sequential MARS approaches proposed in this dissertation. These cases include an inventory forecasting problem, an automotive crash safety design problem and an air pollution SDP problem.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iv

ABSTRACT vi

LIST OF ILLUSTRATIONS xi

LIST OF TABLES xv

List of Algorithms xix

Chapter Page

1. INTRODUCTION 1

  1.1 Motivation 1

  1.2 Research Framework 6

2. LITERATURE REVIEW 7

  2.1 Regression-Based Approaches 8

    2.1.1 MARS 8

    2.1.2 Convex MARS 12

    2.1.3 Polynomial Regression Models 14

    2.1.4 Principal Component Regression 15

    2.1.5 Regression Trees 16

    2.1.6 Projection Pursuit Regression 17

    2.1.7 Artificial Neural Networks 18

    2.1.8 Kernel Regression 19

  2.2 Interpolating Approaches 20

    2.2.1 Kriging 20

    2.2.2 Radial Basis Functions 21

2.3 Summary 21

2.4 Sequential Approaches 25

3. VARIANTS OF MARS 27

3.1 Convex MARS Review 27

3.1.1 Model for Interaction Basis Functions 28

3.1.2 Convexity Proof 30

3.2 Variants of MARS 39

3.2.1 Convex vs. Nonconvex MARS 40

3.2.2 Piecewise-Linear vs. Smooth 42

3.2.3 Piecewise-Linear MARS for Binary Variables 47

4. SEQUENTIAL MARS ALGORITHMS 56

4.1 Sequential MARS 1 - Fit a MARS Function from Scratch. 60

4.2 Sequential MARS 2 - Update the Estimated Model Coefficients. . . . 62

4.3 Sequential MARS 3 - Build on an Existing MARS Function: Sum of MARS Approximations Based on Residuals. 66

4.4 Sequential MARS 4 - Build on an Existing MARS Function: Sum of MARS Approximations. 69

4.5 Sequential MARS 5 - Build on an Existing MARS Function: One MARS Approximation. 81

5. CASE STUDIES 86

5.1 Inventory Forecasting Problem 86

5.1.1 Convex MARS 86

5.1.2 MARS Variants 97

5.2 Safety System Design of Automotive Vehicle: Nonconvex Piecewise-Linear MARS with Binary Variables 102

5.3 Air Quality Problem 114

5.3.1 Sequential MARS 116
5.3.2 Convex MARS Applied in Dynamic Programming 129
6. CONCLUSIONS AND FUTURE WORK 140
Appendix
A. MARS ALGORITHMS 144
REFERENCES 150
BIOGRAPHICAL STATEMENT 162

LIST OF ILLUSTRATIONS

Figure — Page

3.1 Original MARS two-way interaction basis function 27

3.2 Possible directions of a two-way interaction term in original MARS . . 30

3.3 Possible directions of a two-way interaction term in convex MARS when $\phi = +1$ 31

3.4 Possible directions of a two-way interaction terms in convex MARS when $\phi = -1$ 32

3.5 Two-way interaction term in (a) original MARS vs. (b) convex MARS from Shih [1] 42

3.6 Nonconvex (a) vs. convex (b) approximation functions 43

3.7 Two-way interaction term with (a) piecewise-linear fit vs. (b) quintic fit 46

3.8 Directions for a pair of univariate terms for binary variables in piecewise-linear MARS 49

3.9 Possible directions of a two-way interaction terms with binary variables in piecewise-linear MARS when $\phi = +1$ 52

3.10 Possible directions of a two-way interaction terms with binary variables in piecewise-linear MARS when $\phi = -1$ 53

3.11 Possible directions of a two-way interaction terms with a binary variable and a continuous variable with $k_2 = -0.5$ in piecewise-linear MARS when $\phi = +1$ 54

3.12 Possible directions of a two-way interaction terms with a binary variable and a continuous variable with $k_2 = -0.5$ in piecewise-linear MARS when $\phi = -1$ 55

4.1 Sequential MARS approach 59

5.1 Four-dimensional inventory forecasting problem. Comparison of box-plots based on a validation set of 100 points: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$) 89

5.2 Nine-dimensional inventory forecasting problem. Comparison of box-plots based on a validation set of 1000 points: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$) 90

5.3 Four-dimensional inventory forecasting problem. Comparison of box-plots based on a validation set of 100 points. Data sets include random noise, CV=0.01: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$) 97

5.4 Four-dimensional inventory forecasting problem. Comparison of box-plots based on a validation set of 100 points. Data sets include random noise, CV=0.05: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$) 98

5.5 Four-dimensional inventory forecasting problem. Comparison of box-plots based on a validation set of 100 points. Data sets include random noise, CV=0.10: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$) 99

5.6 Nine-dimensional inventory forecasting problem. Comparison of box-plots based on a validation set of 1000 points. Data sets include random noise, CV=0.01: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$) 100

5.7 Nine-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 1000 points. Data sets include random noise, CV=0.05: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$) 101

5.8 Nine-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 1000 points. Data sets include random noise, CV=0.10: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$) 102

5.9 Four-dimensional inventory forecasting problem. Comparison of median absolute error from different data sets (original and with noise, CV=0.01, CV=0.05 and CV=0.10). (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$) 103

5.10 Nine-dimensional inventory forecasting problem. Comparison of median absolute error from different data sets (original and with noise, CV=0.01, CV=0.05 and CV=0.10). (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$) 104

5.11 Four-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 100 points. (1) Original MARS, (2) Nonconvex piecewise-linear, (3) Nonconvex piecewise-linear using original backward algorithm, (4) Nonconvex smooth, (5) Nonconvex smooth using original backward algorithm, (6) Convex piecewise-linear ($\tau > 0$) and (7) Convex smooth ($\tau > 0$) 105

5.12 Nine-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 100 points. (1) Original MARS, (2) Nonconvex piecewise-linear, (3) Nonconvex piecewise-linear using original backward algorithm, (4) Nonconvex smooth, (5) Nonconvex smooth using original backward algorithm, (6) Convex piecewise-linear ($\tau > 0$) and (7) Convex smooth ($\tau > 0$) 106
5.13 Residual plots of the models that showed curvature of the automotive vehicle case 107
5.14 MSE results from the different sequential MARS algorithms 129
5.15 Last period of the air quality SDP optimal value function from Yang et al. [2] 130
5.16 Last period of the air quality SDP optimal solution value function estimated by sequential MARS 1 (iteration 14) (Table 5.24) 131
5.17 Last period of the air quality SDP optimal value function from Ariyajunya [3] 137
5.18 Last period of the air quality SDP optimal solution value function estimated by convex MARS 138

LIST OF TABLES

Table Page

3.1 General parameter settings to generate the variants of MARS 40

3.2 Variants of MARS 46

4.1 Input parameter settings for the initial MARS approximation. 58

4.2 Initial input points for variables $i = 1, ..., n$, $N = 10$. 72

4.3 Selected knots for variables $i = 1, ..., n$. 73

4.4 Existing knots (yellow) and preliminary new selected knots (blue) for variables $i = 1, ..., n$, $N = 20$. 78

4.5 Final knots selection. Existing knots (yellow), preliminary new selected knots (blue) and knots that needed to be relocated (magenta) for variables $i = 1, ..., n$, $N = 20$. 79

5.1 Parameter settings for the versions of MARS on the four-dimensional inventory forecasting problem. 88

5.2 Parameter settings for the versions of MARS on the nine-dimensional inventory forecasting problem. 89

5.3 Main effect terms of the original MARS function for the four-dimensional inventory forecasting problem. 92

5.4 Main effect terms of the original MARS function for the nine-dimensional inventory forecasting problem. 92

5.5 Comparison of various threshold values based on different percentages of the maximum absolute coefficient from original MARS. Median absolute error is reported. 93

5.6 Four-dimensional case. Comparison of various threshold values based on different percentages of the maximum absolute coefficient from original MARS tested in data sets containing random noise. Median absolute error is reported. 94
5.7 Nine-dimensional case. Comparison of various threshold values based on different percentages of the maximum absolute coefficient from original MARS tested in data sets containing random noise. Median absolute error is reported. 95
5.8 Median of the absolute error (four-dimensional case). 96
5.9 Median of the absolute error (nine-dimensional case). 96
5.10 Parameter settings for MARS variants on the four-dimensional inventory forecasting problem. 99
5.11 Parameter settings for MARS variants on the nine-dimensional inventory forecasting problem. 100
5.12 MARS variants absolute error boxplot numerical results on the four-dimensional inventory forecasting problem. 101
5.13 MARS variants absolute error boxplot numerical results on the nine-dimensional inventory forecasting problem. 102
5.14 Description and details of the input variables of the automotive vehicle case. 108
5.15 R-squared of the multiple linear regression models of the automotive vehicle case. 109
5.16 Variance inflation factor of the regressors of the automotive vehicle case.110
5.17 Minimum distance measures between design points. 110
5.18 Minimum distance measures between design points for different input data sets. 111

5.19 Stepwise linear and MARS approximation models for two different data sets (1). 112
5.20 Stepwise linear and MARS approximation models for two different data sets (2). 113
5.21 MARS results for approximating the SDP future value functions and corresponding computational times. 118
5.22 State variables for the last stage of the air quality SDP problem. . . . 120
5.23 General and constant input parameters for sequential MARS approaches. 120
5.24 Sequential MARS 1 applied to the last stage of the SDP air quality problem. 121
5.25 Sequential MARS 2 applied to the last stage of the SDP air quality problem. 122
5.26 Sequential MARS 3 applied to the last stage of the SDP air quality problem. Note: the cumulative number of candidate knots is the number of candidate knots from previous iterations plus the current number of eligible knots; however, to generate the approximation function at each iteration only 10 knots were candidates to choose from. 123
5.27 Sequential MARS 4 applied to the last stage of the SDP air quality problem. 124
5.28 Sequential MARS 5 applied to the last stage of the SDP air quality problem. 125
5.29 Other results for sequential MARS 3 and 4 applied to the last stage of the SDP air quality problem. 126
5.30 Preliminary results for sequential MARS 5 applied to the last stage of the SDP air quality problem. 127

5.31 Cumulative computational times (hr:min:sec) for sequential MARS approaches applied to the last stage of the SDP air quality problem. . . . 128

5.32 MARS results for approximating the SDP future value functions and corresponding computational times. 133

5.33 Results using different thresholds (τ) for modeling the last stage of the SDP air quality problem. 134

5.34 Calculating the threshold value using the proposed approach based on the mean for the inventory forecasting case. 136

5.35 Calculating the threshold value using the proposed approach based on the median for the inventory forecasting case. 136

List of Algorithms

4.1 Sequential MARS 1 Algorithm 63

4.2 Sequential MARS 2 Algorithm 67

4.3 Sequential MARS 3 Algorithm 70

4.4 Sequential MARS 4 Algorithm 71

4.5 Algorithm for setting knots 74

4.6 Algorithm for assigning number of new knots. 76

4.7 Algorithm to determine the final set of knots. 77

4.8 Sequential MARS 5 Algorithm 85

A.1 Convex MARS Interaction Transformation Algorithm (CIT) 147

A.2 Convex MARS Forward Coefficient Restriction Algorithm (FCR) . . . 148

A.3 Convex MARS backward Iteration of Pruning and Refitting Algorithm (BIPR) 149

CHAPTER 1

INTRODUCTION

1.1 Motivation

The purpose of a statistical model is to estimate the relationship between a set of predictor or factor variables and one or more response variables through a mathematical expression and enable the prediction of future responses. Typically, statistical models utilize data that come from observational studies or from designed experiments. In designed experiments, the conditions of the factor variables are controlled, while in observational studies, they are not. However, there are complex situations for which performing a physical experiment is impractical, for example if it is too costly or simply impossible to perform. If we consider the case of identifying control strategies for reducing ozone pollution, it is impractical to create a study that manipulates emissions leading to ozone and then observes the resulting pollution levels. In such cases, it may be possible to develop a computer model to represent the performance of the system, and then use this computer model to study the system.

A simulation is the most common type of computer model used for computer experiments. Often a deterministic simulation is used, but random variation that might exist in the real system can be added by simulating external noise and internal variations in the input variables (Chen et al. [4]). Computer models have been used for decades with many fields of research applying them extensively. Some applications include Vorwerg et al. [5], Wei et al. [6], Drignei and Popescu [7]. For computer model validation see Bayarri et al. [8].

For complex systems, an accurate computer model may be computationally expensive, making it difficult to conduct a comprehensive exploration of the system via a naïve trial and error process. To overcome this difficulty, a statistical model can be used to estimate the relationship between the input and output of the simulation model. This is known as a surrogate model or metamodel of the computer model.

In design and analysis of computer experiments (DACE), metamodel construction has two components, an experimental design and a statistical approximation model. The experimental design is chosen to efficiently explore the input space of the computer model, while the statistical model seeks to provide a computationally quick representation of some aspect of the computer model output, such as a particular performance measure of interest. An important objective of the approximation model is to be able to represent the system accurately. This will definitely rely on the input information given and the objective of the application; therefore, selecting the type of experiment and the statistical model is an important task that will depend on the nature of the system. Sacks et al. [9] emphasizes this in his work and provides an efficient design and analysis of computer experiments (DACE) framework applied to an electronic-circuit simulator example. Other research on computer experiments includes Barton [10], Morris et al. [11], Tu and Barton [12], Barton [13], Wang and Tang [14], Alvarez et al. [15].

The designs of experiments often used for DACE are response surface model (RSM) designs, orthogonal arrays (OA), and Latin hypercube (LH). Other used designs are the ones considered low-discrepancy sequences such as Sobol [16] and Hammersley sequences [17]. Examples of the approximation models are polynomial response surface models, Kriging, Gaussian processes, regression trees, artificial neural network (ANN) and multivariate adaptive regression splines (MARS).

The particular focus of this dissertation is on the MARS method. MARS, introduced by Friedman in 1991 [18] is a flexible statistical modeling technique that can represent nonlinear complex structures with high-dimensional data and interactions. MARS has different important properties that are suitable for both statistical and optimization objectives. It builds the model parsimoniously via stepwise procedures and most importantly, MARS is flexible in its structure that is, the base model can be modified relatively easy. However, it can still be restricted to satisfy certain characteristics required by the underlying function. Other statistical methods such as polynomial models and artificial neural networks present similar properties, in particular flexibility, but their structure is not as easy to manipulate as it is for the MARS model.

Considering this advantage, the MARS method has become very attractive. Different researchers have concentrated their efforts on this method and have modified the original structure seeking to satisfy certain properties (Chen et al. [19], Bakin et al. [20], Tsai and Chen [21], Shih [1] and Weber et al. [22]). The use of the MARS method has been extended in past years and has been applied in diverse research areas such as science, technology, transportation, finance and healthcare. MARS has also been used as a metamodel in DACE approaches (Chen [23], Chen et al. [24], Tsai and Chen [21], Cervellera et al. [25], Siddappa [26], Pilla [27], Yang et al. [2]).

In optimization problems, selecting the right method to use in order to find an optimal solution is a critical task. This method selection primarily depends on the function structure, for example, if it is desired to optimize a function that presents nonconvexity, using an optimization method that can handle nonconvexity is favorable. Now, in situations where the function to optimize needs to be approximated by a statistical model (surrogate method), it is favorable to use an approximation model that can meet the function needs. This motivated the development of different MARS variants that can satisfy different structural properties.

The original MARS method employs a forward and a backward procedure; the variants of MARS follow the same logic to construct the model, however some alterations are made to adapt convexity, linearity and a smoothing routine. By being able to select the model that best corresponds to the underlying function, a more accurate prediction can result from the approximation function. This is advantageous not only for optimization purposes to help finding a local or global optima, but also for statistical purposes where the objective is to obtain the best possible fit.

The variants of MARS use a linear transformation for the interaction terms proposed by Shih [1], who created the convex version of MARS. The quintic function introduced by Chen [28] for smoothing the function is also used in some of the variants. By combining the modifications made by Shih [1] and the smoothing routine, four different variants are derived, convex with a piecewise-linear fit, convex with smoothing routine, nonconvex with a piecewise-linear fit and nonconvex with smoothing routine. The last two have the option to reduce the complexity of the function by selecting the original MARS backward algorithm. An additional advantage is incorporated to the variants that require a piecewise-linear fit, which is the ability to handle binary variables.

The DACE approach has had an enormous contribution for studying complex systems; however, one consistent concern for researchers in optimization is to streamline the process by reducing the computational time. Although computers have become more powerful over the years resulting in a growing use of computer models, optimization of computer models requires running the computer model at many inputs. As researchers seek to study more and more complex systems, corresponding computer models continue to push the limits of computing power.

A strategy to overcome this problem and to reduce the computational time, and thus the computational cost, is to use sequential approaches. A sequential approach refers to the iterative evaluation of a model's performance using a non-fixed sample size that is gradually collected. To quote Fan [29]: "Sequential DACE can be used for metamodel-based design optimization or for global metamodeling (Jin et al. [30])." Sequential design for optimization searches for the optimal set of points for the design while simultaneously exploring other potentially important regions with the purpose of finding a global optimum. On the other hand, sequential approaches for metamodels have the purpose of improving the accuracy at each iteration.

This research introduces efficient sequential algorithms that are based on the MARS model structure. An initial sample size considered to be small is used to obtain a coarse approximation model. Then the sample size is increased at each iteration, enabling the use of current and new points to update the approximation. The accuracy of the model is evaluated at each iteration with a testing data set. The overall approach is then stopped by a criterion that could be based on either statistical or optimization purposes.

The objective of these approaches is to increase the prediction accuracy by updating the approximation function once new data are received. Five different algorithms are proposed:

- Sequential MARS 1 - fit a MARS function from scratch at each iteration,
- Sequential MARS 2 - update the estimated model coefficients at each iteration,
- Sequential MARS 3 - build on an existing MARS function: sum of MARS approximations based on residuals,
- Sequential MARS 4 - build on an existing MARS function: sum of MARS approximations,

- Sequential MARS 5 - build on an existing MARS function: one MARS approximation.

These approaches have been tested on an air pollution problem and have shown efficient results in terms of accuracy and computational time.

1.2 Research Framework

The dissertation is organized as follows. Chapter 2 provides a literature review of different statistical models that are specifically used for high dimensional cases. It also provides a background of existing sequential approaches. Chapter 3 contains an extended review of convex MARS and the explanation of the variants of MARS including the usage of binary variables. Chapter 4 provides the details of the purposed approaches for sequential MARS. Chapter 5 contains three different case studies where both the variants of MARS and sequential MARS approaches are applied. Finally chapter 6 shows the conclusions and future work.

CHAPTER 2

LITERATURE REVIEW

The present section discusses different statistical modeling methods that are well known in the literature; it also distinguishes their main properties. The objective is to review statistical models that can handle high dimensionality with interactions, contain numerical responses, and that can model complex structures. The complexity of a model is the number of degrees of freedom used to fit it (Friedman and Stuetzle [31]).

The different methods can be classified in diverse manners based on their properties, such as parametric vs. nonparametric or regression vs. classification. However the property that is of more interest to this work is the ability to interpolate. Therefore, the classification of the presented methods is divided into two groups, regression-based approaches (which are non-interpolating) vs. interpolating approaches.

Interpolation is used to estimate new values by using the existing information to cover certain regions that are not fully represented in the space by the known values. The estimation must be within the existing ranges. Interpolating may be appropriate in certain occasions; however, there are observed or experimental data for which interpolation should not be used. Some examples would be categorical data or data with noise (unexplained variation or randomness).

The task of a statistical model is to estimate the relationship between input variables and one or more response variables. The mathematical expression that describes this relationship is:

$$Y = f(x) + \varepsilon$$

where $f(x)$ indicates the corresponding value of Y, which is approximated by $\hat{f}(x)$, x is the vector of input variables and ε is the random error, which is assumed to be independent of x and normally distributed ($E(\varepsilon) = 0$). In the data-driven statistical approaches the error distribution is not assumed. For consistency in the terminology, the input variables of a statistical model are called factor variables.

2.1 Regression-Based Approaches

There are many types of regression methods such as linear and nonlinear regression, weighted regression (Atkenson [32]), regression by rule induction (Weiss and Indurkhya [33, 34]), additive models (Stone [35], Wahba [36], Hastie and Tibshirani [37]), among others which can perform well; however, not all the regression models can deal with the course of dimensionality phenomenon. The regression models described below are capable of handling this.

Regression models can be described with the following linear function:

$$\hat{f}(x) = \beta_0 + \sum_{m=1}^{M} \beta_m x_m, \tag{2.1}$$

where $\hat{f}$(x) is the predicted value given by the constant term β_0 and the sum of the linear combination of β_m coefficients multiplied by their corresponding predictor value x_m. M represents the number of regression coefficients.

This section includes the explanation of different regression-based methods, beginning with the description of MARS models; the rest of the models appear in chronological order.

2.1.1 MARS

The MARS statistical method developed by Friedman [18] owns different characteristics that make it suitable for many applications, and for such reason, MARS has gained popularity over the last few years. It is a flexible statistical method, that is, it is capable of fitting nonlinear functions without knowing in advance the properties of such functions. MARS is considered as a nonparametric method (Kuhnert and McClure [38], Psichogios et al. [39], Ben-Ari and Steinberg [40]). It is used for high dimensional data and can handle interaction. MARS can model curvature by bending the function at certain knot locations, which are appropriately selected, and simultaneously choose the most significant variables to be included in the final model. The method involves two main stepwise algorithms: forward algorithm, which selects the possible candidates for the basis functions and backward algorithm, which prunes and refines the model function (the algorithms are shown in appendix A). The approximation model is represented with the following equation:

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

where β_0 is the intercept term, $B_m(x)$ represents the basis functions that depend on the x factor variables, and β_m is the unknown coefficient for such functions. M represents the maximum number of basis function. The basis functions can be univariate or of a higher degree. The interaction terms are created by multiplying an existing basis function by a new univariate term. To represent the univariate terms, MARS uses piecewise-linear functions of the form:

$$b^+(x;k) = [+(x-k)]_+, \;\; or \;\; b^-(x;k) = [-(x-k)]_+, \tag{2.3}$$

where k represents an univariate knot which is selected for each of the factor variables x. The functions are truncated at zero, and thus they only consider the positive side. These functions are known as truncated linear functions or hinge functions. The interaction functions are then represented by:

$$B_m(x) = \prod_{l=1}^{L_m} [s_{l,m} \cdot (x_{v(l,m)} - k_{l,m})]_+ \;\; , \tag{2.4}$$

where the degree order L_m of the interaction term is user defined. The most common it to use two- to three-way interaction terms. It is not recommendable to use higher interaction terms unless the underlying function requires it. This is because MARS method works by partitioning the data into different regions and having higher interaction terms will cause the creation of many regions (basis functions) that can be sensitive to the extreme points. In Equation 2.4, $s_{l,m}$ represents the direction of the univariate term, which could be positive or negative. MARS selects basis function that are added in pairs in the forward algorithm based on generalized cross-validation (GCV) lack-of-fit (LOF) criterion:

$$LOF(\hat{f}_M) = GCV(M)\frac{\sum_{i=l}^{N}(y_i - \hat{f}_M(x_i))^2}{\left(1 - \frac{c(M)}{N}\right)^2} \tag{2.5}$$

where the numerator is the average-squared residual of the fit to the data and the denominator is a penalty associated with the number of basis functions M. In this selection the knots are tested and thus selected. The forward algorithm tends to overfit the data, therefore the backward algorithm is responsible of removing those basis functions that contribute least to the overall model function, in other words, the basis functions that produce the smallest increase in the residual squared error are eliminated. The β_m coefficients are estimated using least squares procedure. Once the basis functions are selected, the truncated linear terms are replaced by a cubic function to provide continuity. The details for the cubic function are presented in Chapter 3 Section 3.2.2.

After the introduction of MARS, various researchers have shown interest suggesting modifications to the method to satisfy certain structural properties. In 1993, Chen [28] proposed a quintic function for smoothing the truncated linear functions. The cubic function used by Friedman has a continuous second derivative everywhere except at the side knots. By using the quintic function, the MARS model has a continuous second derivative throughout the function as well as the region containing the side knots. An example of the MARS model using the quintic function was applied by Chen [23] to estimate the future value function of a SDP inventory forecasting problem. The details for the quintic function are presented in Chapter 3 Section 3.1.2.

In 2000, Bakin et al. [20] proposed the use of second-order B-splines instead of the truncated linear functions (Equation 2.3) with the purpose of numerical stability. In 2005, Tsai and Chen [21] developed automatic stopping rules (ASR) based on the coefficient of determination R^2 and R_a^2 (adjusted) instead of allowing the forward algorithm to reach the maximum number of basis functions ($M_{\max}$). This parameter is selected by the user and as for now, it is basically defined based on a trial-error approach. ASR reduces the computational effort and helps avoiding over-fitting. Additionally, Tsai and Chen implemented a robust version of MARS guaranteeing the selection of lower-order interaction terms over higher-order based on the fit contribution of the function term. It is called robust, since it reduces the sensibility to extreme points. An application includes the estimation of the future value function of a wastewater treatment SDP problem (Tsai and Chen [21]).

Shih in 2006, [1] developed convex versions of MARS transforming the interaction terms to a one-dimensional term and constraining the β_m coefficients to accommodate convexity rules for optimization purposes. In the following section (Section 2.1.2) a detailed explanation of this method is provided.

More recently, Weber et al. [22] proposed an approach called CMARS, where the backward algorithm is modified by constructing a penalized residual sum of squares as a Tikhonov regularization problem, i.e., ridge regression. This is later solved by continuous optimization techniques, in particular conic (C) quadratic programming, although “C” can also stand for convex or continuous. Different data sets were used for evaluating its performance; some of them include metal casting, Parkinson telemonitoring, and red wine quality. Two other studies are derived from CMARS, these are boostraping CMARS (BCMARS) and robust CMARS (RCMARS). In BCMARS proposed by Yazici et al. [41] an empirical distribution is fitted to each parameter of the CMARS model by using a computational method called bootstrap. In RCMARS introduced by Özmen et al. [42] the capability of CMARS is enhanced to be able to handle random input and output variables. A more recent study by Kartal [43] modifies the forward algorithm using a new knot selection procedure based on a mapping approach.

Some studies that include the use of MARS method are Lewis and Stevens [44], Chen [23], Deichmann et al. [45], ShieuMing et al. [46], TianShyug et al. [47, 48], Crino and Brown [49], Yang et al. [2], Pilla et al. [50],

2.1.2 Convex MARS

Convex MARS created by Shih [1] provides the flexibility to model a convex function, even though when the data present some nonconvexity, the algorithms forces a convex fit. To achieve this, two main modifications were implemented in the original MARS; (1) coefficients are constrained, such that pairs of basis functions are guaranteed to jointly form convex functions; (2) the form of interaction terms is altered to eliminate the inherent nonconvexity caused by the product of unviariate terms. The sum of the coefficients needs to be nonnegative, however by restricting this sum by a positive value called threshold (τ), reduces the possibility of convexity violations. Shih proposed to obtain this value by taking a percentage from the maximum absolute coefficient from the original MARS model or from a multiple linear regression model. The percentage range suggested is between 2% and 20%. MARS is then guaranteed to be convex by the fact that the sum of convex functions is convex (Shih [1]). Convex MARS uses three different algorithms (see appendix A for details):

- CIT: Convex Interaction Transformation, creates convex forms of the interaction basis function;
- FCR: Forward Coefficient Restriction, selects model terms; and
- BIPR: Backward Iteration of Pruning and Refitting, prunes model terms.

The original form for the interaction terms (Equation 2.4) is replaced by:

$$B_m(x) = \sum_{l=1}^{L_m} \left[s_{l,m} \cdot \frac{(x_{v(l,m)} - k_{l,m})}{(1 - s_{l,m} k_{l,m})} \right]_+ \tag{2.6}$$

To represent the parent term (existing term) and the split term separately with their corresponding parameters, Equation (2.6) can be detailed as follows:

$$\omega_0(x) = \sum_{l=1}^{L_m - 1} s_{l,m} \cdot \frac{(x_{v(l,m)} - k_{l,m})}{(1 - s_{l,m} k_{l,m})} \quad , \tag{2.7}$$

$$\omega_1(x; \phi_m) = \phi_m \cdot \frac{(x_{v(L_m,m)} - k_{L_m,m})}{(1 - \phi_m \; k_{L_m,m})} \quad . \tag{2.8}$$

Sign ϕ_m (-1 or $+1$) determines two distinct one-dimensional variable directions:

$$z^+(x) = \omega_0(x) + \omega_1(x; \phi_m = +1) \;\; ; \;\; z^-(x) = \omega_0(x) + \omega_1(x; \phi_m = -1) \;\; . \tag{2.9}$$

Equations (2.7) and (2.8), can be re-write:

$$\omega_0(x) = a_{0,m} + \sum_{l=1}^{L_m - 1} a_{l,m} \;\; \cdot x_{v(l,m)}, \tag{2.10}$$

where

$$a_{0,m} = \sum_{l=1}^{L_m - 1} \frac{s_{l,m} \cdot k_{l,m}}{(s_{l,m} \cdot k_{l,m} - 1)} \quad , \quad a_{l,m} = \frac{s_{l,m}}{(1 - s_{l,m} k_{l,m})}. \tag{2.11}$$

and

$$\omega_1(x; \phi_m) = \frac{\phi_m \cdot k_{l,m}}{(\phi_m \cdot k_{l,m} - 1)} + \frac{\phi_m \cdot x_{v(l,m)}}{(1 - \phi_m \cdot k_{l,m})}. \tag{2.12}$$

The pairs of univariate truncated linear functions for convex MARS will then be:

$$b^+(z^+;\tau) = [+(z^+ - \tau)]_+ \quad , \quad b^-(z^+;\tau) = [-(z^+ - \tau)]_+ \quad \text{or} \tag{2.13}$$

$$b^+(z^-;\tau) = [+(z^- - \tau)]_+ \quad , \quad b^-(z^-;\tau) = [-(z^- - \tau)]_+ \ . \tag{2.14}$$

where the multivariate knot k in the original variables x is also transformed and takes the value of $\tau = 0$. The two candidate pairs of interaction basis functions for convex MARS that are considered in the forward algorithm are:

$$B_m(x) = [s_{L,m} = +1 \cdot z^+]_+ \quad , \quad B_{m+1}(x) = [s_{L,m} = -1 \cdot z^+]_+ \quad \text{or} \tag{2.15}$$

$$B_m(x) = [s_{L,m} = +1 \cdot z^-]_+ \quad , \quad B_{m+1}(x) = [s_{L,m} = -1 \cdot z^-]_+ \ . \tag{2.16}$$

Another method that achieves convexity proposed in the literature is the multivariate convex regression with adaptive partitioning (CAP) introduced by Hannah in 2011 [51]. The method consists of fitting linear models within each set of local observations and assures convexity by taking the maximum over supporting hyperplanes.

2.1.3 Polynomial Regression Models

Polynomial models are one of the most traditional methods used in response surface methodology (RSM), which was developed by Box and Wilson in 1951 [52]. One of the first records of design of experiments for polynomial regression is the study of Gergonee in 1815, discussed by Stigler in 1974 [53]. Design of experiments is a main element for RSM, however this part is not discussed in the presented work. Information about this can be seen in Box and Draper [54, 55], Myers and Montgomery [56]. Although the polynomial models are parametric approaches, they can be used to achieve more flexible representations for $f(x)$ using different order degrees; the mathematical representation of a polynomial of degree d is:

$$\hat{f}(x) = \beta_0 + \sum_j \beta_j x_j + \sum_j \sum_{k>j} \beta_{jk} x_j x_k + \sum_j \beta_{jj} x_j^2 + \sum_j \sum_{k>j} \sum_{l>k} \beta_{jkl} x_j x_k x_l + \ldots + \sum_j \beta_{j,j,\ldots,j} x_j^d. \tag{2.17}$$

where β represents the polynomial coefficients and x^d denotes the design variables. On the other hand, a disadvantage is that, if the selected model to represents the data does not coincide with the true structure can yield misleading results. Additionally, when the data include several variables, using a high-degree polynomial may result computational expensive. Another drawback from the optimization perspective is that, cubic or higher-order polynomial models may contain one or more inflection points (Giunta [57]). Regression surfaces are not represented well by low-order polynomials. The use of higher-order polynomials is limited by considerations of sample size and computational feasibility (Friedman and Stuetzle [31]).

2.1.4 Principal Component Regression

Principal component regression (PCR) is usually employed for large data sets that may present correlation. PCR was originated from the principal component analysis (PCA) and in fact, PCR requires the use of PCA in its algorithm. Kendall [58] and Hotelling [59] suggested the use of PCR, these authors' concept relies on the replacement of the factor variables by their principal components. The principal component is a linear combination of the factor variables which explains the variability. PCA employs an orthogonal transformation that is applied to a set of data points of possibly correlated variables to eliminate the correlation. The principal component depends on the scaling of the factor variables, therefore a standardization process should be applied first. Jolliffe [60] remarks various alternatives for the original PCR idea.

2.1.5 Regression Trees

Regression trees approach introduced by Breiman et al. in 1984 [61] is commonly called recursive partitioning or classification and regression trees (CART). It is a similar approach to MARS and in fact, MARS can be viewed as a continuous versions of CART. It is also considered to be flexible and a nonparametric technique (Kuhnert and McClure [38]). Regression trees can represent complex relationship between the factor variables and the response variables without making assumptions about the true function. This method can also identify the variables that contribute the most to the model for representing the relationship of the true function. The idea is to perform a binary recursive partition and split the information into different regions. This method also employs forward and backward algorithms. The forward algorithm selects what variable to split on, and the split point; in the same manner, it constructs the tree topology. The backward algorithm has the purpose to avoid overfitting and captures the important information of the structure. Similarly as other methods, there are parameters that are data driven, in this case the size of the tree should be adaptively chosen by the data. The model can be represented by the linear form presented in Equation 2.2 except that it does not use truncated linear functions, instead it uses step functions, also called indicator functions of the form:

$$b^+(x;k) = 1(x > k), \quad b^-(x;k) = 1(x \le k) \tag{2.18}$$

where k represents the split point, the constant $\beta_0 = 1$ for regression trees. Another difference with respect to MARS is that when an univariate or an interaction existing term is involved in a multiplication by a new term, the existing term is replaced by the new term, and is no longer available for future interactions. This procedure leads to a disadvantage, that is, regression trees find difficult to model additive structures since a node may not split more than once. MARS on the other hand can consider additive effects (Hastie et al. [62]). However, additive models are not the main interest of this research. Another drawback is that the model tends to select terms of high order producing a non-robust model and thus, the model may be difficult to interpret. Interpretability is often seen as an important characteristic.

2.1.6 Projection Pursuit Regression

Projection pursuit regression (PPR) was implemented by Friedman and Stuetzle in 1981 [31]. It is considered as a nonparametric ([31], [40]) multiple adaptive regression method with a successive refinement property. PPR generates the model based on projections of the data using a smoothed representation. It estimates the model by reflecting the training set onto lower dimensional projections as a solution for high-dimensional data sets facing the course of dimensionality problem (Uysal and Güvenir [63]). It has a successive refinement property since it selects the best model at each iteration. The approximation model is represented by the following equation:

$$\hat{f}(x) = \sum_{m=1}^{M} S_{\beta_m}(\beta m \cdot x) \tag{2.19}$$

where M is the number of smooth univariate functions $S_{\beta m}$. The parameter β_m represents the projections for the factor variables x. The algorithm stops selecting terms when the new candidate does not contribute to improving the model fit. It exists flexibility to select the smoothing procedure, however local regression and smoothing splines are recommended. The idea is similar compared to recursive partitioning, however PPR does not split the data, allowing a complex model when necessary; it also allows interaction. PPR model is most useful for prediction, and not very useful for producing an understandable model for the data (Hastie et al. [62]). Research work utilizing projection pursuit involves Mayer [64, 65], Zhang et al. [66], Fu et al. [67], Du et al. [68].

2.1.7 Artificial Neural Networks

Artificial neural networks or neural networks approach was motivated from the actual neural connection from the human brain with the purpose of finding the relationship between the factor variables and response variables, specifically for nonlinear relationships. ANN has been studied for decades initially from the medical perspective. In 1989, White [69] presented neural network from an statistical perspective. ANN is a nonlinear parameterized regression function that can be applied for regression and classification problems. The formulation uses nodes at different layers to interpret the model that are connected via weights. The factor variables are represented by input nodes and the response variables by output nodes. In between their connection, there are hidden nodes that can be in one or more layers and are induced by an activation function. This function can vary depending on the application. Sigmoid and radial basis functions network are the most popular used. The following equation shows the ANN model using sigmoid as the activation function:

$$\sigma(x) = \frac{1}{1+e^{-x}}, \tag{2.20}$$

then the ANN approximation model with one hidden layer is:

$$\hat{f}_k(x; w; v; \theta; \gamma) = \sigma\left(\sum_{h=1}^{H} w_{hk} \cdot Z_h + \gamma k\right) \tag{2.21}$$

where H is the number of hidden nodes, the term w_{hk} links the hidden nodes h to the output nodes k, γk is a constant term called bias node and for each hidden node h, Z_h is equal to the following equation:

$$Z_h = \sigma \left(\sum_{i=1}^{n} v_{ih} \cdot x_i + \theta_h \right) \tag{2.22}$$

where n is the number of factor variables, the term v_{ih} links the input nodes i to the hidden nodes h, and θ_h is also a bias node. An important decision to take when using ANN is the structure of the model, that is, the activation function, and other parameters such as the number of hidden layers. By increasing the number of hidden layers, the complexity of the structure increases as well as the computational effort. ANN method is considered to be flexible; and similarly as other statistical methods, it can provide a high-precision fit on the training data but a poor fit on new predictions if the model parameters are not properly selected. Research work that includes the use of ANN can be seen in Cervellera et al. [25].

2.1.8 Kernel Regression

Kernel regression is a nonlinear, nonparametric technique for estimating regression functions from noisy data (Bishop [70]). Kernel methods use weights that decrease smoothly to zero with distance from the target point; and in high-dimensional spaces the distance kernels are modified to emphasize some variables more than others (Hastie et al. [62]). In Kernel regression, the data can come from a fixed design or random design. A common kernel (weighting function) interpolator is the Nadaraya-Watson estimator ([71, 72]), which has the form of a normalized expansion in Gaussian radial basis functions defined in the input space (Bishop [70]). The selection of the smoothing function is critical for having a good fit. Studies in the literature that use kernel methods include Cervellera et al. [73], Dharmasena et al. [74].

Mühlenstädt and Kuhnt [75] introduced a Kernel interpolation method using inverse distance weighting with a piecewise-linear function, where the basic idea is to combine many locally fitted linear functions in order to obtain an interpolator. Kernel interpolation is an alternative to Kriging (Section 2.2.1) for situations with small sample sizes and situations with non-stationary behavior (Mühlenstädt and Kuhnt [75]).

2.2 Interpolating Approaches

2.2.1 Kriging

Kriging also known as a spatial correlation model is a continuous interpolating and flexible method that was originated in geostatistics by Matheron in 1963 [76]. It was introduced into the modeling and computer experiments by Sacks et al. in 1989 [9] and later studied by Santner et al. [77] from a Bayesian perspective. This method can be seen as an adaption of least square regression to the interpolation task (Mühlenstädt and Kuhnt[78]). It assumes a correlation structure and it uses this correlation to predict responses values between observed points (Chen et al. [4]). The general form of the model is:

$$Y(x) = \sum_{m=1}^{M} \beta_m b_m(x) + Z(x) \tag{2.23}$$

where b_m is the known function for each unknown β_m coefficients, and $Z(x)$ is the random process commonly assumed to be Gaussian. A major advantage of Kriging over other interpolating methods is that, it provides an uncertainty estimate that can be used for judging the local fit of the interpolator (Mühlenstädt and Kuhnt [75]). Modifications to Kriging have been proposed by Li and Sudjianto [79], Xiong et al. [80] to deal with small sample sizes and Hoessjer and Hartman [81] for large sample sizes using Gaussian Markov random fields.

2.2.2 Radial Basis Functions

Radial Basis Function (RBF) is a data-driven method, which is considered as a tool for the interpolation of multidimensional scattered data. Radial basis functions has been extensively studied by Powell since 1987 [82]. It consists of the sum of scalar parameters λ_i combined with n number of radial basis functions θ. The function approximation is of the form:

$$Y(x) = \sum_{i=1}^{n} \lambda_i \theta(\| x - x_i \|) \;\; x \in R^d \tag{2.24}$$

where each radial basis function is related with a norm $\| \cdot \|$, which is typically represented by the Euclidean distance between the basis points (or center points) x_i, at which a function is known, and the points x, at which the approximation is evaluated. Other distance functions can also be employed. Each RBF is also associated with a coefficient λ_i that represents its weight. This parameter can be estimated using linear least squares. Examples of radial univariate basis functions are linear, cubic and thin plate splines functions. Other examples of radial basis functions can also include parametric basis functions such as Gaussian, inverse quadric, multiquadric, and inverse multiquadric. The selection of the precise shape of the radial basis function is important to determine how good or bad the approximation is. This method can be considered flexible since it has the ability to fit a variety of different functions. and thus, an advantage of it is that it can be applied to almost any dimensional set of data; Powell [83] studied the RFB for functions of many variables, Buhmann [84] provided a theoretical analysis of radial basis functions and included implementations.

2.3 Summary

Previous sections have presented a general review of different statistical methods divided into two different categories, regression-based approaches and interpolating approaches. The approaches included in the former category are MARS and its variants, polynomial regression, PCR, regression trees, PPR, ANN and Kernel regression; while the later category presented RBB, Kriging and a variant of a Kernel for interpolation. The question arises as to which method is most appropriate to use; unfortunately there is no straightforward answer to that. The number of options could be reduced after knowing more about the system where it is desired to be used. The more information about the underlying function, the more advantageous. However, there is no perfect model, a method can be just as good as the input data.

There are many comparisons and review papers that focus on the properties and performance of certain methods. For non-interpolating approaches, Psichogios et al. [39] compared MARS and neural networks; both methods showed promising results for fitting general nonlinear multivariate functions; MARS however, seemed to be more accurate and faster than neural networks. Conlin et al. [85] reviewed different statistical projection techniques: PCA, PCR, and projection to latent structures (PLS); and nonlinear modeling techniques: MARS and ANN. Uysal et al. [63] provided an overview of regression techniques for knowledge discovery and compared different methods, where rule-based regression and MARS outperformed other methods such as CART. Chen et al. [4] offered a review of design and modeling in computer experiments with extensive research of previous works. Muñoz and Felicísimo [86] compared the performance of logistic multiple regression (LMR), PCA, CART and MARS applied to two different data sets, where MARS and CART achieved the best prediction success. Leathwick et al. [87] made a comparative study of generalized additive models and MARS showed a strong performance. Ben-Ari and Steinberg [40] compared Kriging, MARS and PPR on high-dimensional data finding Kriging as the best performer.

For interpolating approaches, Mühlenstädt and Kuhnt [78] compared different interpolation methods on two dimensional test functions; the methods included were Kriging, thin plate spline, natural neighbor interpolation and Kernel interpolation. The results showed that no method is always preferable for every situation and that the performance can also depend on the experimental design.

More generally, Jin et al. [88] made an extensive comparative study evaluating the performance of polynomial regression, MARS, RBF and Kriging using fourteen test problems. The comparison was made based on accuracy and robustness criteria using large and small sample sets. He also talked about the computational effort for constructing the models and pointed out that the experimental design may have some influence on the methods performance.

Jin et al. [88] recommended to compare the performance of the methods based on multiple metrics such as accuracy, efficiency, robustness, model transparency and simplicity.

For any statistical estimation method, an important metric is its accuracy, which can be seen from the fitting accuracy on the training data and from the accuracy on future predictions. However, one cannot say that a technique is superior to others in terms of accuracy, since a certain method may not work well for specific problems. It is also true that the more complex the structure, the higher the computational effort, and studies make this comparison; however it may not be fair to compare if the programming code or software used is not the same for all the methods. Another reason that may affect the selection of a method is the level of popularity of the method. This may be due to the limitations on the computer's capacity by the time the methods were originated or perhaps the availability of commercial software to construct certain models.

In summary, in order to achieve better results, the experimental design and the approximation method should be cautiously selected based on the user criterion; and a comparison with at least two different methods is recommended. From the optimization perspective, it is possible to reduce the alternatives. To quote Jin et al. [88]: “The computational cost of complex simulation makes it impractical to rely exclusively on simulation codes for the purpose of design optimization.” A strategy to overcome this problem is the usage of metamodels. There are some models that are more suitable to use for optimization. A regression-based approach is preferable over interpolating models since they can be sensitive to the noise. The risk of using interpolating models for optimization is that, the search of the optimization method may be limited and thus, may not explore all regions for finding the optimal solution. If the interpolation is based on existing points that are not close to the optimal point, the space is discarded by the optimization search.

From the models that do not interpolate, flexibility is the next most important property preferred for optimization. Polynomial models and ANN may present different alternatives that conduce flexibility, however their structure is not as easy to modify as it is for the MARS model; this may be the case as well for PCR and PPR. Additionally PPR might be difficult to interpret. The regression trees structure may be modified, however in high-dimensional problems robustness is relevant; and regression trees technique may not always provide a robust model since it tends to select higher order terms. The options are limited to Kernel and MARS. These two methods seem to have more potential for optimization based on surrogate models.

Studies that compare metamodels include Booker et al. [89] who presented a framework for generating a sequence of approximations to the objective functions and managing the use of these approximations as surrogates optimization. A review of metamodeling applications can be seen in Barthelemy and Haftka [90], Sobieszczanski-Sobieski and Haftka[91].

2.4 Sequential Approaches

Sequential DACE approaches can be developed in the sampling (design of computers experiments) phase or in the metamodeling phase. Sequential approaches are iterative methods that use fewer data points than conventional practices that use all the generated sample points. The purpose is to sequentially update the model once new data are received and stop when it is deemed that there is adequate information to represent the system.

The idea in sequential experimental designs is to strategically select the points based on optimal criteria and update the model. In sequential metamodeling the objective is to improve the accuracy of the model by updating the approximation function once new data are received. These data have no particular order.

One of the earliest sequential design approaches is a simple two-stage optimal design produced by the integrated mean squared error criterion by Sacks et al. [9] where kriging was used as the metamodel. Other sequential design approaches are given by Osio and Amon [92], who proposed the use of nonlinear regression models as surrogates fitted with data coming from a deterministic numerical simulation using optimal sampling based on Bayesian approaches and A-optimal criteria. Jin et al. [30] proposed an efficient algorithm for constructing optimal design of computer experiments. The method employs an enhanced stochastic evolutionary algorithm and efficient methods for evaluating different optimality criteria. This work was concentrated but not limited to optimizing Latin Hypercube designs.

Other numerous studies have been made on sequential designs Huang et al. ([93], Christen and Sansó [94], Crombecq et al. [95] and Quan et al. [96]). Researchers have been focused on sequential design approaches using different techniques for selecting the optimal points and using different metamodels, however approaches for sequential metamodeling are lacking in the research literature.

Fan [29] developed sequential algorithms for feed-forward neural networks to both identify the approximating model structure and determine sample size. The algorithm builds an adaptive value function approximation within a dynamic program while the size of the state space sample grows. He employed low-discrepancy sequence sampling techniques to increment the sampling of the state space.

CHAPTER 3

VARIANTS OF MARS

3.1 Convex MARS Review

Convexity for the purposes of optimization refers to a function that is either purely convex or purely concave, since a concave function can be made convex by multiplying by a negative scalar. If a function simultaneously has convex and concave structure, then this is considered to be a nonconvex function, which is different to optimize. Original MARS follows this structure in its algorithms, that is it compensates the function approximation with both convex and nonconvex basis functions; therefore, the existence of nonconvexity is possible. However, for an underlying convex function an approximation only of convex univariate pairs in theory should be possible using the original MARS. Figure 3.1 illustrates a two-way interaction term from the original MARS that clearly shows nonconvexity.

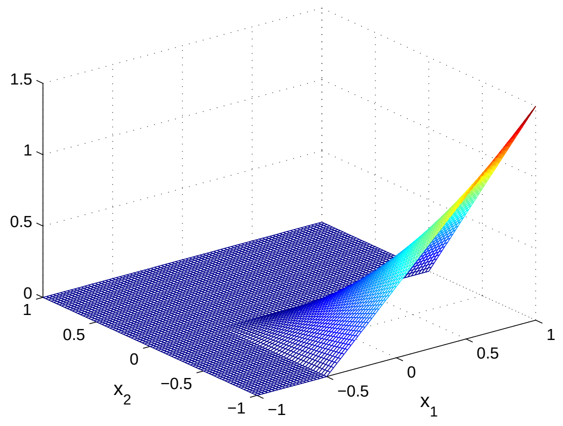


Figure 3.1. Original MARS two-way interaction basis function.

The following proofs demonstrate that convexity and nonconvexity might be present in an original MARS two-way interaction term. Considering the interaction term from Figure 3.1, $b(x) = [+(x_1 + 0.5)]_+ \cdot [-(x_2 - 0)]_+$, $b(x)$ is computed from different selected points to show such cases. To prove nonconvexity:

$A : x_1 = 1, x_2 = -0.2\ ;\ b(x_A) = 0.3$,

$B : x_1 = -0.2, x_2 = -1\ ;\ b(x_B) = 0.3$, and a point between these ranges:

$C : x_1 = 0, x_2 = -0.7\ ;\ b(x_C) =$0.35.

By linear interpolation, we estimate $\hat{b}(x_C)$,

$$\hat{b}(x_C) = b(x_A) + \frac{b(x_B) - b(x_A)}{x_{2B} - x_{2A}} \cdot (x_{2C} - x_{2A}) = 0.3$$

By having $b(x_C) = 0.35 > \hat{b}(x_C) = 0.3$, shows that the plot is concave at point C. Similarly, to show convexity:

$D : x_1 = 0, x_2 = -0.5\ ;\ b(x_D) = 0.25$,

$E : x_1 = 0, x_2 = 0.5\ ;\ b(x_E) = 0$, and a point between these ranges:

$F : x_1 = 0, x_2 = 0\ ;\ b(x_F) =$0.

By linear interpolation, we estimate $\hat{b}(x_F)$,

$$\hat{b}(x_F) = b(x_D) + \frac{b(x_E) - b(x_D)}{x_{2E} - x_{2D}} \cdot (x_{2F} - x_{2D}) = 0.125$$

By having $b(x_F) = 0 < \hat{b}(x_F) = 0.125$, shows that the plot is convex at point F. In response to the nonconvexity issue, convex MARS was developed guaranteeing convexity.

3.1.1 Model for Interaction Basis Functions

The initial approach for convex MARS approximation adds the parameter ϕ to provide additional different options for the direction of the interaction term being the possibilities Equations (2.15) and (2.16) previously explained in Chapter 2 in

Section 2.1.2. However to better represent the structure of original MARS, the parameters ϕ and s are swapped. An adjusted and refined formulation for the interaction terms is shown below:

$$B_m(x) = [\phi_m \cdot z_m]_+. \tag{3.1}$$

where, ϕ_m can be either + or -, and z_m can take the positive z_m^+ or negative z_m^- side. z_m is defined as:

$$z_m = a_{0,m} + \sum_{l=1}^{L_m} a_{l,m} \cdot x_{v(l,m)}, \tag{3.2}$$

where,

$$a_{0,m} = \sum_{l=1}^{L_m} \frac{s_{l,m} \cdot k_{l,m}}{(s_{l,m} \cdot k_{l,m} - 1)} \quad , \quad a_{l,m} = \frac{s_{l,m}}{(1 - s_{l,m} k_{l,m})}. \tag{3.3}$$

Following the updated notation, the candidate pairs for interaction basis functions are:

$$B_m(x) = [\phi_m = +1 \cdot z^+]_+ \quad , \quad B_{m+1}(x) = [\phi_m = -1 \cdot -z^+]_+ \quad \text{or} \tag{3.4}$$

$$B_m(x) = [\phi_m = +1 \cdot z^-]_+ \quad , \quad B_{m+1}(x) = [\phi_m = -1 \cdot -z^-]_+ \ . \tag{3.5}$$

The following plots (Figure 3.2) show the possible direction for the interaction terms of original MARS. The knot value for x_1 is $k_{1,m} = 0.25$ and $k_{2,m} = -0.5$ for x_2. The two upper plots have $s_{2,m} = +1$, and $s_{1,m} = -1$ and $s_{1,m} = +1$ respectively. In the same manner, the lower plots set $s_{2,m} = -1$, and $s_{1,m} = -1$ and $s_{1,m} = +1$ respectively. An example of a candidate two-way interaction pair would be the left upper plot and the right lower plot.

By switching the parameters ϕ and s, the directions for interaction terms are represented in Figures (3.3) and (3.4). The values used for the knots are the same as in Figure 3.2. Figure 3.3 represents the possible directions when $\phi = +1$ and Figure 3.4 when $\phi = -1$.

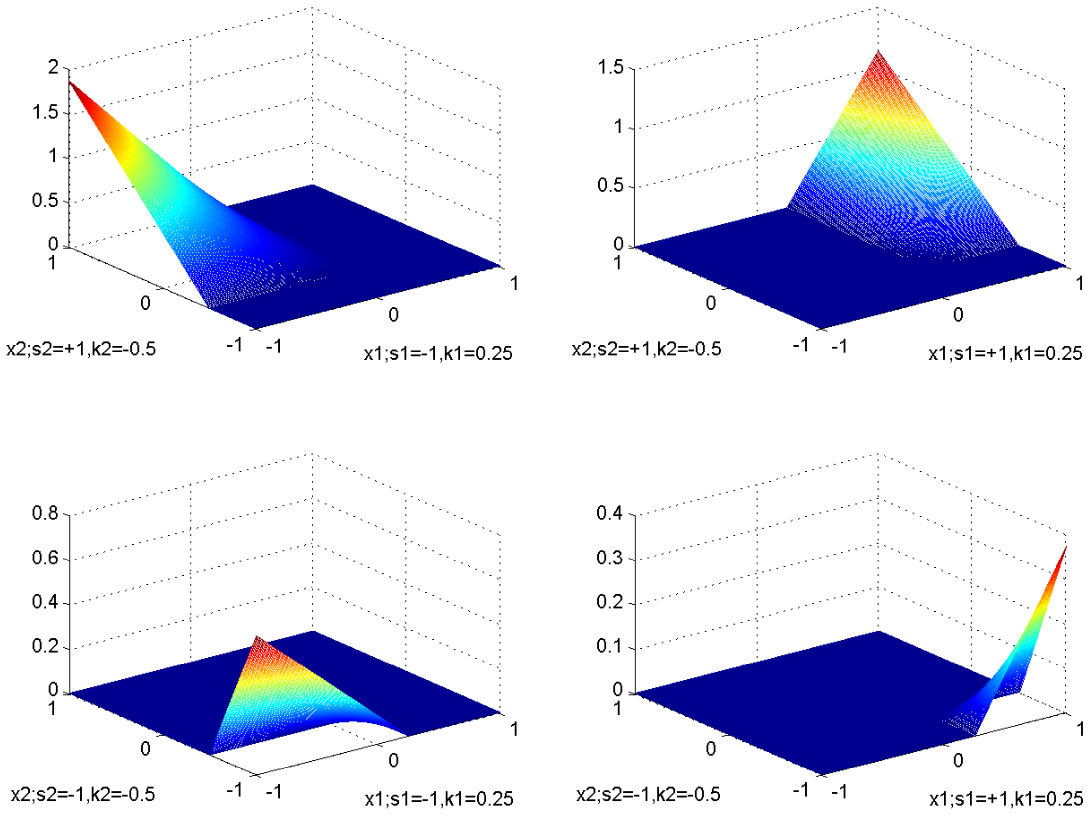


Figure 3.2. Possible directions of a two-way interaction term in original MARS.

3.1.2 Convexity Proof

To guarantee convexity in the approximation, the β_m coefficients for the basis functions are constrained and the interaction basis functions are transformed to a one-dimensional term. However, the transformed univariate truncated linear terms (Equations 2.13 and 2.14) employ a quintic function to smooth their shape after they are selected to be in the final model. Therefore a convexity proof for the pairs

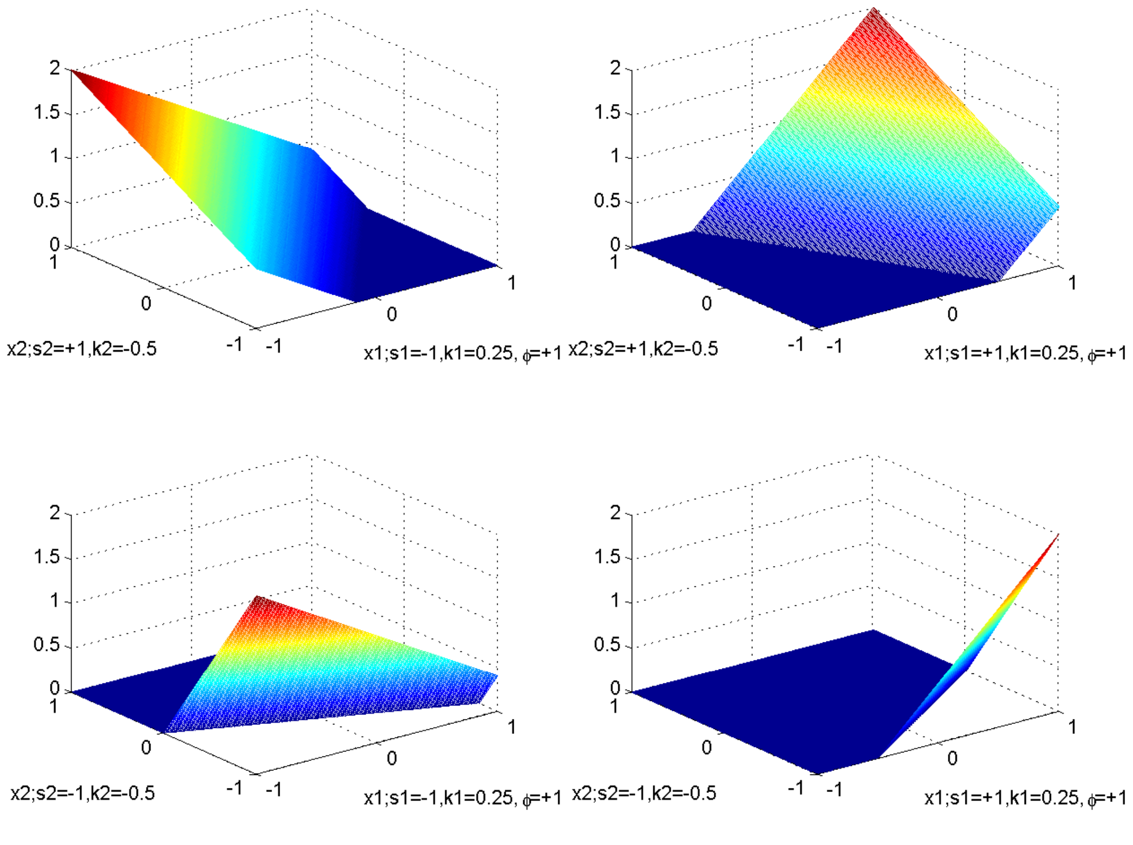


Figure 3.3. Possible directions of a two-way interaction term in convex MARS when $\phi = +1$.

of convex MARS univariate terms is required. We first show the original quintic function derived by Chen [28]:

$$Q(x|s=+1,k_-,k,k_+) = \begin{cases} 0, & x \le k_- \\ \alpha_+(x-k_-)^3 + \beta_+(x-k_-)^4 + \gamma_+(x-k_-)^5, & k_- < x < k_+ \\ x-k, & x \ge k_+, \end{cases} \tag{3.6}$$

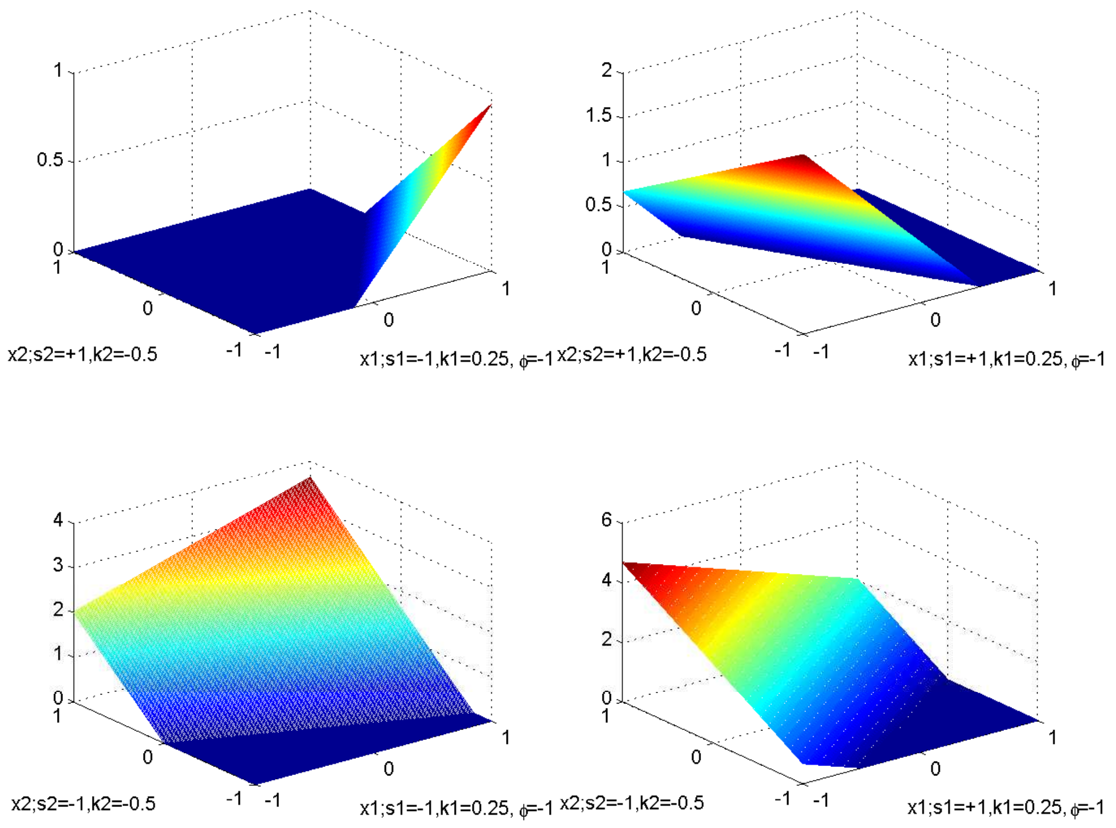


Figure 3.4. Possible directions of a two-way interaction terms in convex MARS when $\phi = -1$.

where,

$$\alpha_+ = \frac{6k_+ - 10k + 4k_-}{(k_+ - k_-)^3},$$
$$\beta_+ = \frac{-8k_+ + 15k - 7k_-}{(k_+ - k_-)^4},$$
$$\gamma_+ = \frac{3k_+ - 6k + 3k_-}{(k_+ - k_-)^5},$$

and

$$Q(x|s=-1,k_-,k,k_+) =$$
$$\begin{cases} k-x, & x \leq k_- \\ \alpha_-(x-k_+)^3+\beta_-(x-k_+)^4+\gamma_-(x-k_+)^5, & k_- < x < k_+ \\ 0, & x \geq k_+, \end{cases} \tag{3.7}$$

where,

$$\alpha_- = \frac{(-1)(4k_+ - 10k + 6k_-)}{(k_- - k_+)^3},$$
$$\beta_- = \frac{(-1)(-7k_+ + 15k - 8k_-)}{(k_- - k_+)^4},$$
$$\gamma_- = \frac{(-1)(3k_+ - 6k + 3k_-)}{(k_- - k_+)^5}.$$

Without loss of generality, a center knot of zero is specified. And to simplify the proof, the knots can be defined in terms of deltas as follow: $\Delta = k_+ - k_-$, $\Delta_1 = k_+ - k$, and $\Delta_2 = k - k_-$. Then the equalities from the quintic functions (Equations 3.6 and 3.7) respectively can be written as:

$$\alpha_+ = \frac{[6\Delta_1 - 4\Delta_2]}{\Delta^3},$$
$$\beta_+ = \frac{[-8\Delta_1 + 7\Delta_2]}{\Delta^4},$$
$$\gamma_+ = \frac{[3\Delta_1 - 3\Delta_2]}{\Delta^5},$$

and

$$\alpha_- = \frac{[4\Delta_1 - 6\Delta_2]}{\Delta^3},$$
$$\beta_- = \frac{[7\Delta_1 - 8\Delta_2]}{\Delta^4},$$
$$\gamma_- = \frac{[3\Delta_1 - 3\Delta_2]}{\Delta^5}.$$

The forward algorithm adds terms in pairs, so the purpose of the convex proof is to show that the sum of the univariate terms:

$$Q = \beta_1 Q(x|s=+1, k_-, k, k_+) + \beta_2 Q(x|s=-1, k_-, k, k_+)$$

is a convex function on $[k_-, k_+]$, where β_1 and β_2 are the unknown coefficients of the basis functions. A twice differentiable function is convex if and only if its second derivative is ≥ 0. To prove that the quintic function from Equations (3.6) and (3.7) is a convex function on $[k_-, k_+]$, the second derivative of Q with respect to x is calculated. Let be:

$$Q_1 = Q(x|s=+1, k_-, k, k_+); \;\; Q_2 = Q(x|s=-1, k_-, k, k_+) \tag{3.8}$$

The first derivative of Q_1 and Q_2 is:

$$Q_1{}' = \left(\frac{3[6\Delta_1 - 4\Delta_2][(x-k_-)^2]}{\Delta^3} + \frac{4[-8\Delta_1 + 7\Delta_2][(x-k_-)^3]}{\Delta^4}\right) + \left(\frac{5[3\Delta_1 - 3\Delta_2][(x-k_-)^4]}{\Delta^5}\right).$$

and

$$Q_2{}' = \left(\frac{3[4\Delta_1 - 6\Delta_2][(x-k_+)^2]}{\Delta^3} + \frac{4[7\Delta_1 - 8\Delta_2][(x-k_+)^3]}{\Delta^4}\right) + \left(\frac{5[3\Delta_1 - 3\Delta_2][(x-k_+)^4]}{\Delta^5}\right).$$

Second derivative is then:

$$Q_1{}'' = \left(\frac{6[6\Delta_1 - 4\Delta_2][(x-k_-)]}{\Delta^3} + \frac{12[-8\Delta_1 + 7\Delta_2][(x-k_-)^2]}{\Delta^4}\right) + \left(\frac{20[3\Delta_1 - 3\Delta_2][(x-k_-)^3]}{\Delta^5}\right). \tag{3.9}$$

and

$$Q_2{}'' = \left(\frac{6[4\Delta_1 - 6\Delta_2][(x-k_+)]}{\Delta^3} + \frac{12[7\Delta_1 - 8\Delta_2][(x-k_+)^2]}{\Delta^4}\right) + \left(\frac{20[3\Delta_1 - 3\Delta_2][(x-k_+)^3]}{\Delta^5}\right). \tag{3.10}$$

Nonconvexities are produced in the quintic basis functions when the center knot k is not close enough to the midpoint between k_- and k_+ [28]. To avoid such nonconvexities, we must constrain

$$\frac{\Delta_1}{\Delta} \geq \frac{2}{5} \text{ and } \frac{\Delta_2}{\Delta} \geq \frac{2}{5}. \tag{3.11}$$

Since it is possible to have different scenarios for the β_m coefficients:

- $\beta_1 + \beta_2 \geq 0$ and
- $\beta_1 + \beta_2 = 0$.

The following cases are derived:

CASE I:

$$\beta_1 + \beta_2 \geq 0;\;\; \beta_2 > -\beta_1;\;\; \beta_2 = -\beta_1 + d, \text{ where } d \geq 0.$$

so we need to prove that

$$Q'' = \beta_1 Q_1'' + (-\beta_1 + d)Q_2'' \geq 0. \tag{3.12}$$

also, it is constrained to:

$$\frac{\Delta_1}{\Delta} = \frac{2}{5} \text{ and } \frac{\Delta_2}{\Delta} = \frac{3}{5}. \tag{3.13}$$

First, by substituting deltas in Equations (3.9) and (3.10), ${Q_1}''$ and ${Q_2}''$ can be simplified to:

$${Q_1}'' = \left(\frac{12[(x-k_-)^2]}{\Delta^3} - \frac{12[(x-k_-)^3]}{\Delta^4}\right). \tag{3.14}$$

and

$${Q_2}'' = \left(\frac{-12[(x-k_+)]}{\Delta^2} - \frac{24[(x-k_+)^2]}{\Delta^3} - \frac{12[(x-k_+)^3]}{\Delta^4}\right). \tag{3.15}$$

Then, by plugging Equations (3.14) and (3.15) into Equation 3.12 we have:

$$Q'' = \beta_1 \left(\frac{12[(x-k_-)^2]}{\Delta^3} - \frac{12[(x-k_-)^3]}{\Delta^4} \right) -$$
$$\beta_1 \left(\frac{12[(x-k_+)]}{\Delta^2} + \frac{24[(x-k_+)^2]}{\Delta^3} + \frac{12[(x-k_+)^3]}{\Delta^4} \right) +$$
$$d \left(\frac{12[(x-k_+)]}{\Delta^2} + \frac{24[(x-k_+)^2]}{\Delta^3} + \frac{12[(x-k_+)^3]}{\Delta^4} \right).$$

which in order to assure convexity need to be Q'' is ≥ 0. To simplify, let $\alpha = \frac{12}{\Delta^3}$ then we have,

$$Q'' = \frac{\beta_1 \alpha}{\Delta} \left((x-k_-)^2[\Delta - (x-k_-)] + (x-k_+)[\Delta^2 + 2\Delta(x-k_+) + (x-k_+)^2] \right) -$$
$$\frac{d\alpha}{\Delta} \left((x-k_+)[\Delta^2 + 2\Delta(x-k_+) + (x-k_+)^2] \right).$$

By substituting $\Delta = k_+ - k_-$ and simplifying we have,

$$Q'' = -d\alpha \left(\frac{(x-k_+)(x-k_-)^2}{(k_+ - k_-)} \right)$$

And by substituting $\alpha = \frac{12}{\Delta^3}$,

$$Q'' = 12d \left(\frac{-(x-k_+)(x-k_-)^2}{(k_+ - k_-)^4} \right)$$

Since $k_- < x < -k_+$, $(x-k_+) < 0$, $(x-k_-) > 0$, and $(k_+ - k_-) > 0$ and since $d > 0$, then Q'' is always > 0, therefore Q is a convex function.

CASE II:

$$\beta_1 + \beta_2 = 0; \quad \beta_2 = -\beta_1.$$

Thus, we need to prove that $Q'' = Q_1'' - Q_2'' \geq 0$. Equation (3.11) holds, and without loss of generality we assume that $\Delta = 1$. Therefore,

$$\Delta = \Delta_1 + \Delta_2 = k_+ - k + k - k_- = k_+ - k_-$$

$$\Delta_1 = \Delta - \Delta_2 = k_+ - k_- - (k - k_-) = k_+ - k$$

$$\Delta_2 = \Delta - \Delta_1 = k_+ - k_- - (k_+ - k) = k - k_-$$

We replace $\Delta_2 = \Delta - \Delta_1$ in Equation (3.9),

$$Q_1'' = 12\left(\frac{[5\Delta_1 - 2\Delta][(x-k_-)]}{\Delta^3} + \frac{[-15\Delta_1 + 7\Delta][(x-k_-)^2]}{\Delta^4}\right) + \left(\frac{[10\Delta_1 - 5\Delta][(x-k_-)^3]}{\Delta^5}\right). \tag{3.16}$$

Similarly for Equation (3.10),

$$Q_2'' = 12\left(\frac{[5\Delta_1 - 3\Delta][(x-k_+)]}{\Delta^3} + \frac{[15\Delta_1 - 8\Delta][(x-k_+)^2]}{\Delta^4}\right) + \left(\frac{[10\Delta_1 - 5\Delta][(x-k_+)^3]}{\Delta^5}\right). \tag{3.17}$$

We need to prove that $Q'' \geq 0$. By plugging Equations (3.16) and (3.17) and simplifying, the following equation can be written:

$$Q'' = 12\left((5\Delta_1 - 2)(x-k_-) + (-15\Delta_1 + 7)(x-k_-)^2 + (10\Delta_1 - 5)(x-k_-)^3\right) - 12\left((5\Delta_1 - 3)(x-k_+) + (15\Delta_1 - 8)(x-k_+)^2 + (10\Delta_1 - 5)(x-k_+)^3\right). \tag{3.18}$$

which can be reduced to:

$$Q'' = 12(5\Delta_1)\left((x-k_-) - 3(x-k_-)^2 + 2(x-k_-)^3 - (x-k_+) - 3(x-k_+)^2 - 2(x-k_+)^3\right) + 12\left(-2(x-k_-) + 7(x-k_-)^2 - 5(x-k_-)^3 + 3(x-k_+) + 8(x-k_+)^2 + 5(x-k_+)^3\right). \tag{3.19}$$

Let $f = (x - k_-)$ and $(f-1) = (x - k_+)$, so Equation (3.19) can be represented as follows:

$$Q'' = 12(5\Delta_1)\left(f - 3f^2 + 2f^3 - (f-1) - 3(f-1)^2 - 2(f-1)^3\right) + 12\left(-2f + 7f^2 - 5f^3 + 3(f-1) + 8(f-1)^2 + 5(f-1)^3\right). \tag{3.20}$$

By solving Equation (3.20) results that $Q'' = 0$, therefore, Q is convex on $[k_-, k_+]$.

CASE III:

$$\beta_1 + \beta_2 = 0; \quad \beta_2 = -\beta_1;$$

Similarly, as case II, we need to prove that $Q'' = Q_1'' - Q_2'' \geq 0$. Also, it is constrained to Equation (3.13): From Equations (3.14) and (3.15), we have

$$Q'' = \left(\frac{12[(x-k_-)^2]}{\Delta^3} - \frac{12[(x-k_-)^3]}{\Delta^4}\right) +$$
$$\left(\frac{12[(x-k_+)]}{\Delta^2} + \frac{24[(x-k_+)^2]}{\Delta^3} + \frac{12[(x-k_+)^3]}{\Delta^4}\right).$$

By replacing $\alpha = \frac{12}{\Delta^3}$ we have,

$$Q'' = \frac{\alpha}{\Delta}\left((x-k_-)^2[\Delta - (x-k_-)] + (x-k_+)[\Delta + (x-k_+)]^2\right)$$

By substituting $\Delta = k_+ - k_-$, we have,

$$Q'' = \frac{\alpha}{\Delta}(x^2k_+ - 2xk_-k_+ + k_-^2k_+ - x^3 +$$
$$2x^2k_- - xk_-^2 - x^2k_+ + 2xk_-k_+ - k_-^2k_+ + x^3 - 2x^2k_- + xk_-^2). \tag{3.21}$$

By solving Equation (3.21), we have that $Q'' = 0$ guaranteeing convexity for Q.

CASE IV:

$$\beta_1 + \beta_2 \geq 0; \;\; \beta_2 > -\beta_1; \;\; \beta_2 = -\beta_1 + d, \text{ where } d \geq 0.$$

so we need to prove that

$$Q'' = \beta_1 Q_1'' + (-\beta_1 + d)Q_2'' \geq 0.$$

Without loss of generality we assume that $\Delta = 1$. And from Equation (3.11) we constrain

$$\frac{2}{5} \leq \frac{\Delta_1}{\Delta} \leq \frac{3}{5}. \tag{3.22}$$

From Equation (3.18) the following equation can be written:

$$Q'' = 12\beta_1\left((5\Delta_1 - 2)(x-k_-) + (-15\Delta_1 + 7)(x-k_-)^2 + (10\Delta_1 - 5)(x-k_-)^3\right) -$$
$$12\beta_1\left((5\Delta_1 - 3)(x-k_+) + (15\Delta_1 - 8)(x-k_+)^2 + (10\Delta_1 - 5)(x-k_+)^3\right) +$$
$$12d\left((5\Delta_1 - 3)(x-k_+) + (15\Delta_1 - 8)(x-k_+)^2 + (10\Delta_1 - 5)(x-k_+)^3\right).$$

which can be reduced to:

$$Q'' = 12d\left((5\Delta_1 - 3\Delta)(x-k_+) + [15\Delta_1 - 8\Delta][(x-k_+)^2] + [10\Delta_1 - 5\Delta][(x-k_+)^3]\right).$$

By substituting $f = (x - k_-)$ and $(f-1) = (x - k_+)$, we have,

$$Q'' = 60d(1-f)\{f[(\Delta_1 - \frac{2}{5}) + f^2(1-2\Delta_1)]\},$$

Since $d \geq 0$ and $0 \leq f \leq 1$, proving $Q'' \geq 0$ requires showing:

$$(\Delta_1 - \frac{2}{5}) + f(1-2\Delta_1) \geq 0. \tag{3.23}$$

Under the constraint in (Equation 3.22), the left-hand side of (Equation 3.23) is minimized at $f = 1$ and $\Delta_1 = \frac{3}{5}$, at which it is equal to zero. Hence, Equation (3.23) holds, and Q is convex on $[k_-, k_+]$.

3.2 Variants of MARS

This section explains the advantages of having different variants for MARS. MARS provides a smooth, flexible approximation to an unknown function in multiple dimensions. Even though MARS method does not require knowing any information about the true function, there are specific applications that assume certain behavior, for example a cost function is known to be convex or on the other hand, a profit function is known to be concave. Therefore, to obtain a more accurate approximation of the function under study, it is suitable to have a statistical method that best fits the underlying function properties.

The existing C-programming algorithm that generates the original MARS written by Chen [28] and which includes a quintic function for smoothing the approximation has been modified. First, Tsai and Chen [21] incorporated the automatic stopping rules for the forward algorithm and then, Shih [1] created the convex MARS version. Now, this algorithm has been adapted to support different options to model the function according to its properties. These options are convex vs. nonconvex and piecewise-linear vs. smooth. These different versions of MARS can be developed by selecting the appropriate input parameters. Table 3.1 describes the necessary settings to develop these different MARS variants.

Table 3.1. General parameter settings to generate the variants of MARS

Variants of MARS	**Parameter settings**		
	convex	smooth	alg3
Nonconvex piecewise-linear	0	0	0
	0	0	1
Nonconvex smooth	0	1	0
	0	1	1
Convex piecewise-linear	1	0	0
Convex smooth	1	1	0

3.2.1 Convex vs. Nonconvex MARS

In many optimization methods, one of the features that is required is convexity, and such characteristic is not a typical assumption of statistical modeling methods. An optimization problem can be simply defined as a function $f(x)$ where the objective can be minimization or maximization, subject to some constraints defined as $x \in S$ where, S represents the feasible region as a set in $\Re^n$. A convex function is known to be convex if and only if all the points that lie in the function and above are a convex set, that is, it contains all the convex combination of its elements. A function f: $C \rightarrow \Re$, where C is a convex set in $\Re^n$, is called convex if

$$f(\lambda \cdot x_1 + (1-\lambda)x_2) \geq \lambda \cdot f(x_1) + (1-\lambda)f(x_2). \tag{3.24}$$

for any x_1, $x_2 \in C$ and $0 \geq \lambda \geq 1$. The convex functions have continuity and differentiability properties that are convenient for optimization methods. If the function f is convex as well as the set S we can assure that every local minimum is a global minimum. As in the opposite for nonconvex functions, the expectation is to have many local minimum points. When there is a function that is known to be convex, it is then desirable to have an approximation method that can handle convexity. Convex MARS, explained in previous Sections 2.1.2 and 3.1 guarantees to estimate $\hat{f}$ a convex function allowing the use of convex optimization. On the other hand, if the function to be estimated is known to be nonconvex, the selection of the optimization method may be more flexible since there are many optimization approaches for such functions. However, it is not guaranteed to find the global optimum. A nonconvex function can be estimated by using the same linear transformation for the interaction terms (Equation 3.2) thus, the function is piecewise-linear. A nonconvex piecewise-linear approximation may or may not select the usage of the original backward algorithm depending on the overall objective. By selecting the algorithm, a less complex structure of the function is selected to represent the model. Alternatively, when the computational execution time is a bigger concern than the high precision of the approximation model, then not selecting the backward algorithm may be beneficial. Figure 3.5 from Shih [1] illustrates a two-way interaction term for an original MARS approximation and for a convex MARS approximation. The nonconvexity in the original MARS version is clearly visible while the convex MARS option eliminates it. Figure 3.6 represents a nonconvex approximation (a) and a convex approximation (b). The graphs were generated using different data sets coming from the same application, where the data present some nonconvexities issues but the underlying function is known to be convex. Convex versions of MARS force a convex fit even when the data are not completely convex. The nonconvex versions still represent nonconvexity in the function.

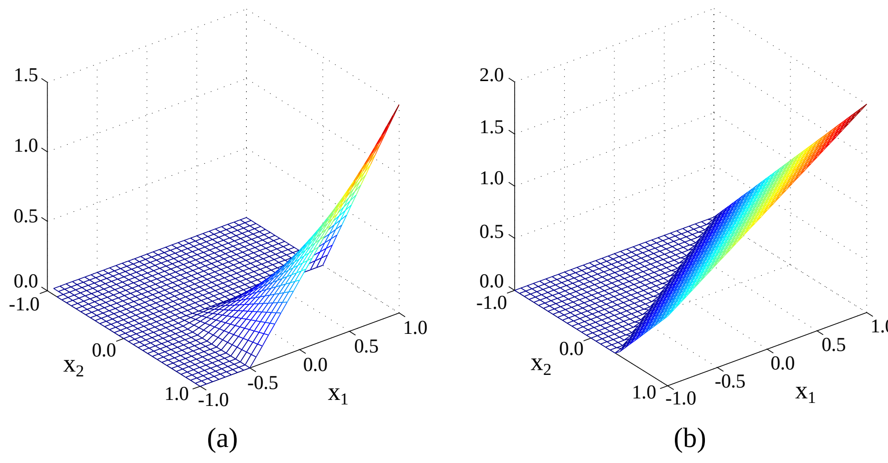


Figure 3.5. Two-way interaction term in (a) original MARS vs. (b) convex MARS from Shih [1].

3.2.2 Piecewise-Linear vs. Smooth

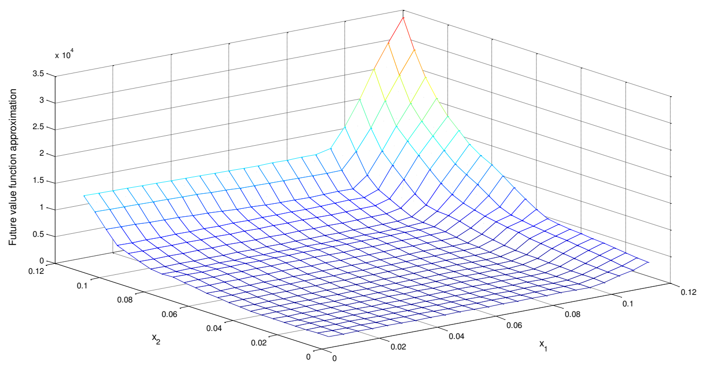
The original MARS approximation method [18] uses a cubic function to smooth the truncated piecewise-linear functions (Equation 2.3), which can be also represented as follows:

$$[s(x-k)]_+ \tag{3.25}$$

The cubic function is:

$$C(x|s=+1,k_-,k,k_+) = \begin{cases} 0, & x \le k_- \\ \alpha_+(x-k_-)^2+\beta_+(x-k_-)^3, & k_- < x < k_+ \\ x-k, & x \ge k_+, \end{cases} \tag{3.26}$$

(a)



(b)

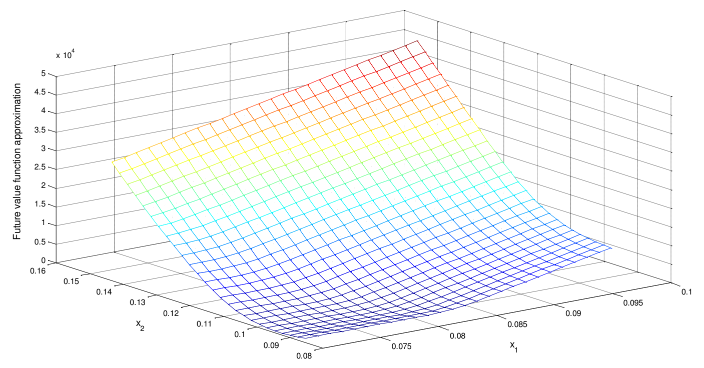


Figure 3.6. Nonconvex (a) vs. convex (b) approximation functions.

where,

$$\alpha_+ = \frac{2k_+ - 3k + k_-}{(k_+ - k_-)^2},$$
$$\beta_+ = \frac{-k_+ + 2k - k_-}{(k_+ - k_-)^3},$$

and

$$C(x|s=-1,k_-,k,k_+) = \begin{cases} k-x, & x \le k_- \\ \alpha_-(x-k_+)^2 + \beta_-(x-k_+)^3, & k_- < x < k_+ \\ 0, & x \ge k_+, \end{cases} \tag{3.27}$$

where,

$$\alpha_- = \frac{(-k_+ + 3k - 2k_-)}{(k_- - k_+)^2},$$
$$\beta_- = \frac{(k_+ - 2k + k_-)}{(k_- - k_+)^3}.$$

The cubic function has only one continuous derivative at the side knots. Chen [23] purposes the use of a quintic function (Equations 3.6 and 3.7) for smoothing option, assuring two continuous derivatives at the side knots. Similarly as in the case of the quintic function, the cubic function can also suffer from nonconvexity issues if the knots are not constrained to certain limits:

$$\frac{k_+ - k}{k_+ - k_-} \ge \frac{1}{3} \text{ and } \frac{k - k_-}{k_+ - k_-} \ge \frac{1}{3}. \tag{3.28}$$

The objective of using either the cubic or the quintic routine is to provide continuity to the multiple piecewise-linear basis functions selected to represent the overall MARS approximation model. In this manner the final model assures a smooth shape. The MARS variants described in this section only use the quintic function when a smoothing procedure is required. When there is no need to smooth the function, a piecewise-linear approximation is possible since the linear transformation is used to select the basis functions. In a piecewise-linear approximation, only the center knot is needed, that is:

$$L(x|s=+1,k,) : (x-k)_+, \;\; and$$

$$L(x|s = -1, k,) : (k - x)_+,$$

Figure 3.7 represents the sum of univariate terms (a pair) using the piecewise-linear function (a) and the quintic function (b) considering that the values of the β_m coefficient are $\beta_1 = \beta_2 = 1$.

The quintic and the piecewise-linear function can be used in both, convex and nonconvex MARS versions. An optimization advantage for a function that presents smoothness is that it can use gradient-base optimization methods. While a piecewise-linear function permits the use of linear programming methods for optimization.

MARS variants provide more flexibility to estimate the function based on the underlying true function and thus facilitate the optimization process. Table 3.2 shows the different variants of MARS and the algorithms required to develop them. Nonconvex piecewise-linear uses the forward coefficient restriction (FCR) algorithm, mentioned in Section 2.1.2, which includes the linear transformation for the interaction terms (Equation 3.2) but the convexity restrictions for the β_m coefficients of the basis functions are disabled. Nonconvex piecewise-linear has also the option of selecting the original MARS backward algorithm for pruning the model. The nonconvex smooth version is similar as nonconvex piecewise-linear, except that it uses the quintic function to provide continuity to the approximation model. Convex piecewise-linear and convex smooth utilize the forward coefficient restriction (FCR) algorithm including the transformation of the interaction terms to one-dimensional terms and the restriction of the β_m coefficients for the basis functions. Additionally, they both need the backward iteration of pruning and refitting (BIPR) algorithm that performs the final check for convexity on the coefficients. Convex piecewise-linear does not use any smooth option, while convex smooth uses the quintic function previously described.

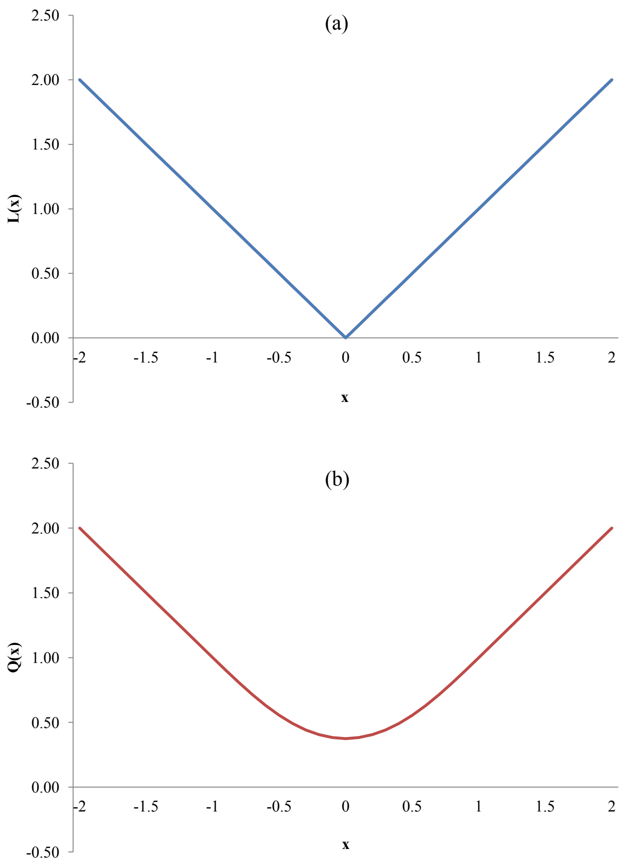


Figure 3.7. Two-way interaction term with (a) piecewise-linear fit vs. (b) quintic fit.

Table 3.2. Variants of MARS

Variants of MARS	**FCR**		**BIPR**	Original Backward Algorithm
	Linear transformation	Coefficients restriction		
Nonconvex piecewise-linear	x			
	x			x
Nonconvex smooth	x			
	x			x
Convex piecewise-linear	x	x	x	
Convex smooth	x	x	x	

3.2.3 Piecewise-Linear MARS for Binary Variables

Original MARS can handle categorical variables by using binary variables. This capability has been incorporated to the piecewise-linear variants of MARS. Input variables can only take the values of -1 or $+1$, therefore the only possible values for the knots are these values. For that reason when the input data involve binary variables a smoothing option cannot be applied since the quintic function needs the center and the side knots while the piecewise-linear function only needs the center knot.

When the variable $x_{v(l,m)}$ is binary, if the sign is negative ($s_{l,m} = -1$), the only possible value for the knot is positive ($k_{l,m} = +1$); alternatively, if the sign is positive ($s_{l,m} = +1$), the only possible value for the knot is negative ($k_{l,m} = -1$). This is because if the sign and the knot of the same variable have the same direction, the denominator in the linear transformation formula (Equation 3.2) would be zero, resulting an undefined division.

Another condition is that in a two-way interaction term one of the variables must be continuous otherwise the interaction term is not significant. The reason is because having a two-way interaction term that contains only binary variables is equivalent to adding two main effects terms of binary variables.

The following proof represents this case. From the truncated linear function:

$$[s_{l,m} \cdot (x_{v(l,m)} - k_{l,m})]_+ \tag{3.29}$$

and the linear transformation (Equation 3.2), we have:

$$\left[\phi\left(\frac{s_{1,m} \cdot k_{1,m}}{(s_{1,m} \cdot k_{1,m} - 1)} + \frac{s_{1,m} \cdot x_{1(1,m)}}{(1 - s_{1,m} k_{1,m})} + \frac{s_{2,m} \cdot k_{2,m}}{(s_{2,m} \cdot k_{2,m} - 1)} + \frac{s_{2,m} \cdot x_{2(2,m)}}{(1 - s_{2,m} k_{2,m})}\right)\right]_+ = [s_{1,m} \cdot (x_{1(1,m)} - k_{1,m})]_+ + [s_{2,m} \cdot (x_{2(2,m)} - k_{2,m})]_+ \tag{3.30}$$

For the interaction term, it is possible to have different cases:

I. $x_{1(1,m)}$, $k_{l,m} = -1$, $s_{1,m} = +1$ and $x_{2(2,m)}$, $k_{2,m} = +1$, $s_{2,m} = -1$

II. $x_{1(1,m)}$, $k_{l,m} = -1$, $s_{1,m} = +1$ and $x_{2(2,m)}$, $k_{2,m} = -1$, $s_{2,m} = +1$

III. $x_{1(1,m)}$, $k_{l,m} = +1$, $s_{1,m} = -1$ and $x_{2(2,m)}$, $k_{2,m} = +1$, $s_{2,m} = -1$

IV. $x_{1(1,m)}$, $k_{l,m} = +1$, $s_{1,m} = -1$ and $x_{2(2,m)}$, $k_{2,m} = -1$, $s_{2,m} = +1$, equivalent to I.

From Equation (3.30) and case I, we have,

$$\left[\phi\frac{1}{2}(1 + x_1 + 1 - x_2)\right]_+ = C[(x_1 + 1)_+ + (1 - x_2)_+] \tag{3.31}$$

The constant C on the right hand side of the equation is added to compensate the left hand side and make it equivalent. The standardization of the $x_{v(l,m)}$ variables when creating the MARS function is based on the midrange and half rage which guarantees that the scaled values are always within the range of $[-1, 1]$. Therefore,

$$(x_1 + 1 + 1 - x_2)_+ = (x_1 + 1)_+ + (1 - x_2)_+,$$

which we can call constant D. If $\phi = 1$,

$$\frac{1}{2} \cdot D = C \cdot D \text{ and } C = \frac{1}{2}. \tag{3.32}$$

Otherwise, if $\phi = -1$,

$$\left[-\frac{1}{2} \cdot D\right]_+ \neq C \cdot D \text{ and } C = \frac{1}{2}. \tag{3.33}$$

Additionally, when $\phi = -1$ the direction that is created by the interaction term is a flat surface at zero, which is irrelevant to the overall approximation function. The above proof applies for the rest of the cases, thus we can conclude that an interaction term using only binary variables is not significant.

Figure 3.8 represents an univariate pair of binary variables. Figures 3.9 and 3.10 show the directions of a two-way interaction term considering only binary variables when $\phi = +1$ and when $\phi = -1$ respectively. From these figures, it can be observed that having only binary variables is not meaningful and that one continuous variable must be involved in the interaction term. Figures 3.11 and 3.12 illustrate a two-way interaction term considering one binary variable (x_1) and one continuous variable (x_2) with a knot value of $k_2 = -0.5$.

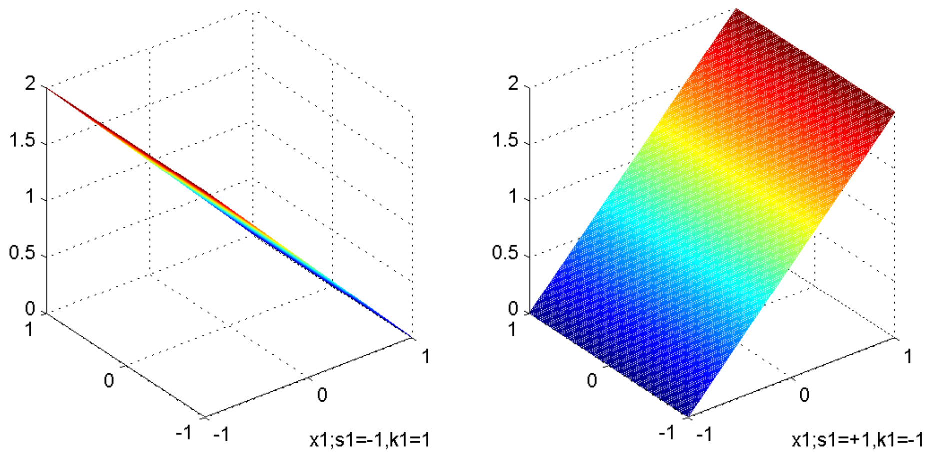


Figure 3.8. Directions for a pair of univariate terms for binary variables in piecewise-linear MARS.

Now, in order to define if one or more binary variables should be considered in a higher order interaction term, a similar proof was developed for a three-way interaction term. From the truncated linear function (Equation 3.29) and the linear transformation (Equation 3.2), we have:

$$\Bigg[\phi\Bigg(\frac{s_{1,m}\cdot k_{1,m}}{(s_{1,m}\cdot k_{1,m}-1)}+\frac{s_{1,m}\cdot x_{1(1,m)}}{(1-s_{1,m}k_{1,m})}+\frac{s_{2,m}\cdot k_{2,m}}{(s_{2,m}\cdot k_{2,m}-1)}+\frac{s_{2,m}\cdot x_{2(2,m)}}{(1-s_{2,m}k_{2,m})} \\ +\frac{s_{3,m}\cdot k_{3,m}}{(s_{3,m}\cdot k_{3,m}-1)}+\frac{s_{3,m}\cdot x_{3(3,m)}}{(1-s_{3,m}k_{3,m})}\Bigg)\Bigg]_+ = \\ [s_{1,m}\cdot(x_{1(1,m)}-k_{1,m})]_+ + [s_{2,m}\cdot(x_{2(2,m)}-k_{2,m})]_+ + [s_{3,m}\cdot(x_{3(3,m)}-k_{3,m})]_+ \quad (3.34)$$

If we let $x_{1(1,m)}$ and $x_{2(2,m)}$ be binary variables and $x_{3(3,m)}$ continuous with a knot value of $-1 < k_{3,m} < 1$, the following cases are possible:

I. $k_{l,m} = -1$, $s_{1,m} = +1$; $k_{2,m} = +1$, $s_{2,m} = -1$; $-1 < k_{3,m} < 1$, $s_{3,m} = -1$

II. $k_{l,m} = -1$, $s_{1,m} = +1$; $k_{2,m} = -1$, $s_{2,m} = +1$; $-1 < k_{3,m} < 1$, $s_{3,m} = -1$

III. $k_{l,m} = +1$, $s_{1,m} = -1$; $k_{2,m} = +1$, $s_{2,m} = -1$; $-1 < k_{3,m} < 1$, $s_{3,m} = -1$

IV. $k_{l,m} = +1$, $s_{1,m} = -1$; $k_{2,m} = -1$, $s_{2,m} = +1$; $-1 < k_{3,m} < 1$, $s_{3,m} = -1$, equivalent to I.

The same cases but considering a positive sign for the continuous term ($s_{3,m} = +1$) are also possible. All cases can use positive or negative direction for the term ϕ. From Equation (3.34) and case I with $\phi = +1$, we have,

$$\left[\frac{1}{2}(x_1+1)+\frac{1}{2}(1-x_2)+\frac{1}{1+k_3}(k_3-x_3)\right]_+ = [(x_1+1)]_+ + [(1-x_2)]_+ + [(k_3-x_3)]_+ \quad (3.35)$$

If the result for the continuous variable x_3 in Equation 3.35 is zero, that is, if

$$(k_3 - x_3) = 0,$$

then the right and left hand side of Equation 3.35 are equivalent since from before we have that $(x_1+1+1-x_2)_+ = (x_1+1)_+ + (1-x_2)_+$ and that the constant $C = \frac{1}{2}$ (used to compensate the left hand side).

$$\left[\frac{1}{2}(1+x_1+1-x_2)\right]_+ = C[(x_1+1)+(1-x_2)]_+$$

If the result for the continuous variable x_3 in Equation 3.35 is positive that is, if

$$(k_3 - x_3) > 0,$$

and considering the previous proof, the Equation 3.35 can be rewritten as follows:

$$\left[\frac{1}{2}(1 + x_1 + 1 - x_2)\right]_+ + \left[\frac{1}{1+k_3}(k_3 - x_3)\right]_+ = C[(x_1 + 1) + (1 - x_2)]_+ + [(k_3 - x_3)]_+ \tag{3.36}$$

To simplify, let $D = (x_1 + 1 + 1 - x_2)_+ = (x_1 + 1)_+ + (1 - x_2)_+$ and plug the value of the constant C,

$$\left[\frac{1}{2} \cdot D\right]_+ + \left[\frac{1}{1+k_3}(k_3 - x_3)\right]_+ = \frac{1}{2} \cdot D + [(k_3 - x_3)]_+ \tag{3.37}$$

If we compensate the left hand side of the Equation 3.37 by adding a constant $E = \frac{1}{1+k_3}$ on the right hand side, both sides are then equivalent.

$$\left[\frac{1}{2} \cdot D\right]_+ + \left[\frac{1}{1+k_3}(k_3 - x_3)\right]_+ = \frac{1}{2} \cdot D + E \cdot [(k_3 - x_3)]_+ \tag{3.38}$$

However if the result for the continuous variable x_3 in Equation 3.35 is negative that is, if

$$(k_3 - x_3) < 0,$$

the result of the left hand side is still considered in the summation while the result in the right hand side is truncated to zero, therefore, the equation is not equivalent.

Since we have at least one case where both sides of the equation are not equivalent, then we can conclude that a three-way interaction term is not the same as summing two binary main effects and one continuous main effect. Therefore, in a three-way interaction term or higher order only one binary variable is allowed.

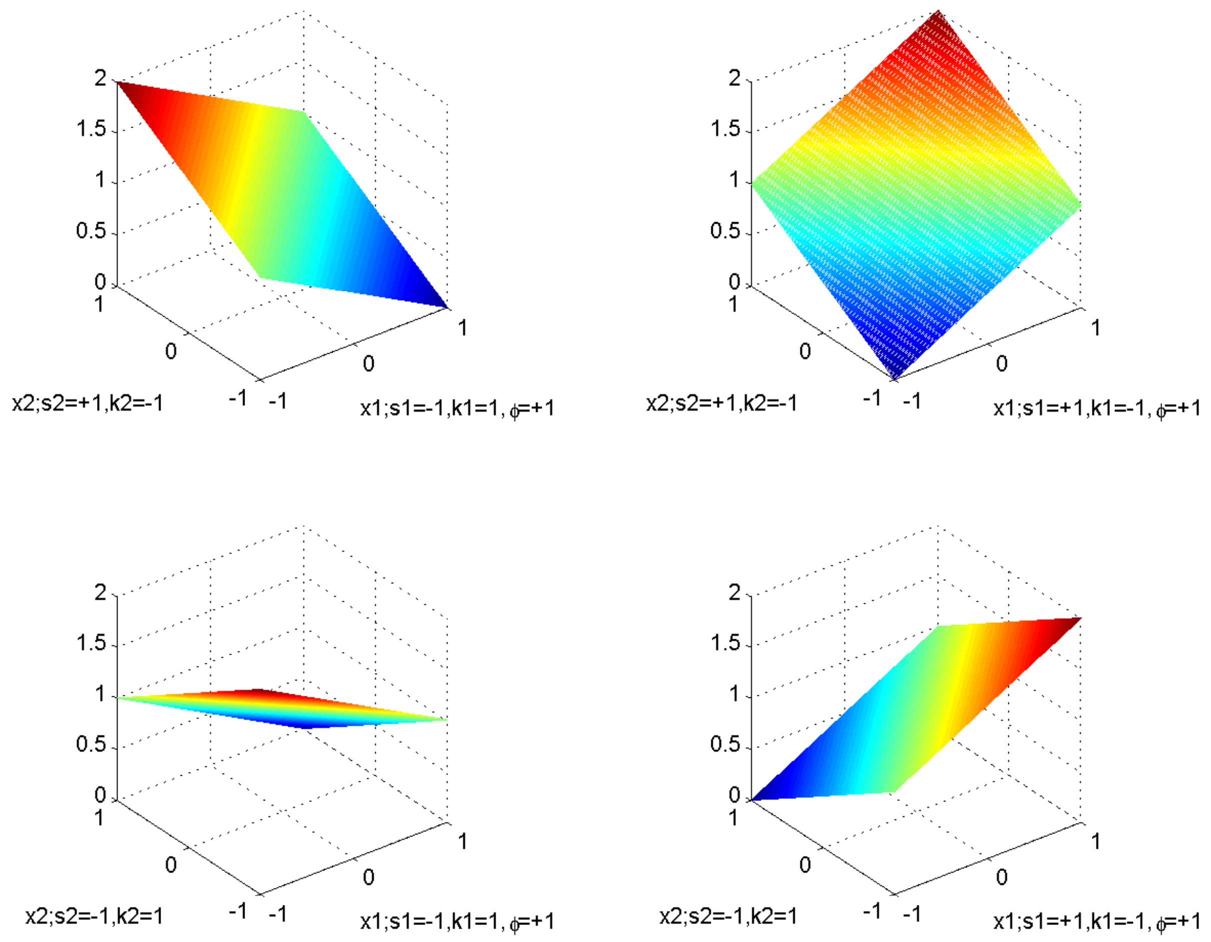


Figure 3.9. Possible directions of a two-way interaction terms with binary variables in piecewise-linear MARS when $\phi = +1$.

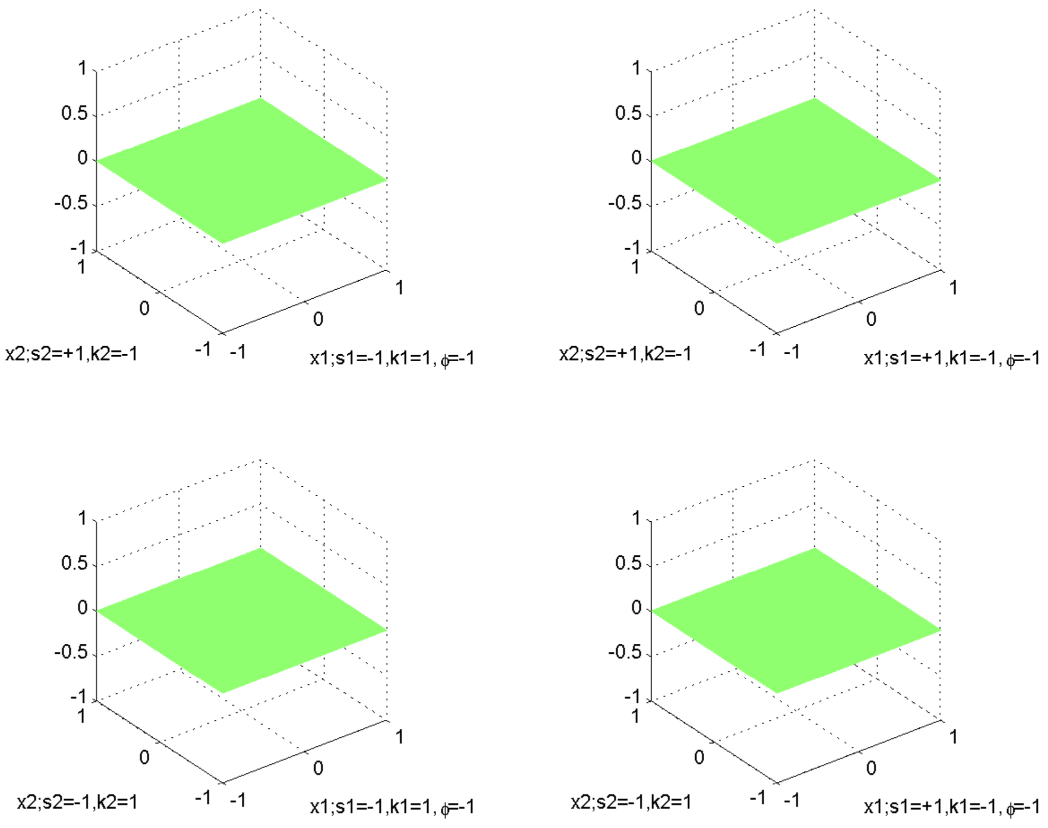


Figure 3.10. Possible directions of a two-way interaction terms with binary variables in piecewise-linear MARS when $\phi = -1$.

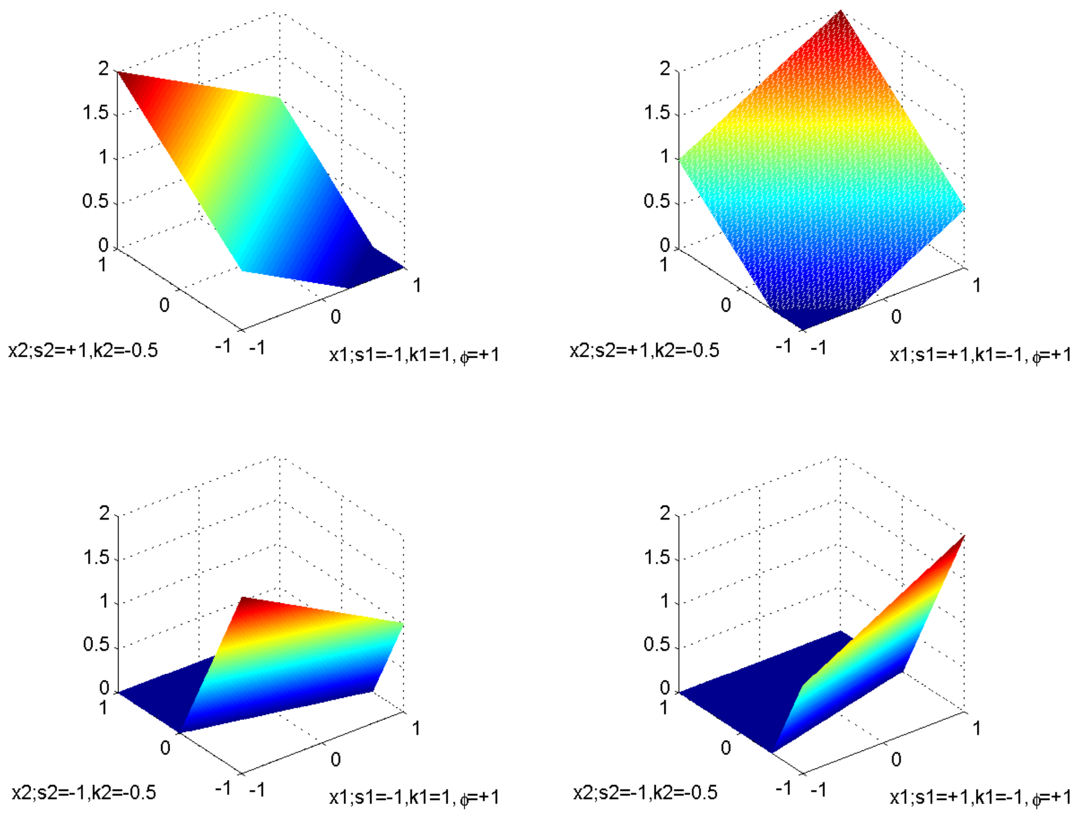


Figure 3.11. Possible directions of a two-way interaction terms with a binary variable and a continuous variable with $k_2 = -0.5$ in piecewise-linear MARS when $\phi = +1$.

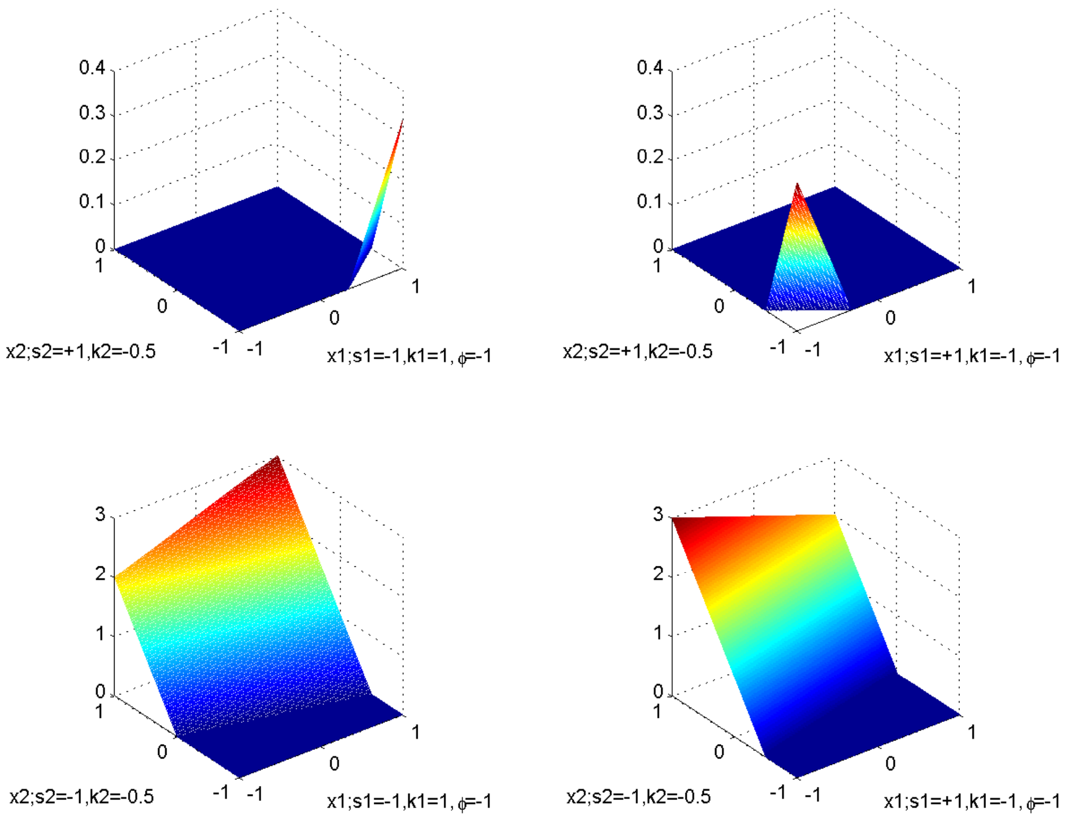


Figure 3.12. Possible directions of a two-way interaction terms with a binary variable and a continuous variable with $k_2 = -0.5$ in piecewise-linear MARS when $\phi = -1$.

CHAPTER 4

SEQUENTIAL MARS ALGORITHMS

The purpose of sequential MARS is to obtain an accurate approximation function by slowing growing the number of sample points and sequentially updating the approximation function. This eliminates the needs to pre-specify an adequate sample size and MARS approximation settings. As a consequence, computational effort could be reduced and a sufficiently accurate approximation may be guaranteed. An initial set of input data points is first employed to obtain a coarse approximation. The number of initial data points is considered to be small. The MARS function is fitted and the performance of the function is evaluated. Then a sequential method is selected. The different options are:

- Sequential MARS 1 - fit a MARS function from scratch at each iteration,
- Sequential MARS 2 - update the estimated model coefficients at each iteration,
- Sequential MARS 3 - build on an existing MARS function: sum of MARS approximations based on residuals,
- Sequential MARS 4 - build on an existing MARS function: sum of MARS approximations,
- Sequential MARS 5 - build on an existing MARS function: one MARS approximation.

Once the selected method is performed, the function is again evaluated to see if it meets the desired requirements that are determined by the application. The sequential MARS algorithm stops when the model considers that the data are reasonable enough to represent the system. These stopping criteria will depend on the application. Sequential MARS can be used with different purposes, statistical or optimization focused. For optimization purposes, if a function needs to be estimated, then sequential MARS can be embedded within an optimization routine. The MARS approximation model can be used as the metamodel to approximate an objective or other function within the optimization. This approximated function is later employed directly in an optimization routine. An automated sequential algorithm is needed for practical implementation within optimization. It is important to mention that the time required for estimating the MARS function depends absolutely on the complexity of the model. Some parameters, such as the maximum number of basis functions $M_{\max}$ and the number of dimensions n, are critical to determine the speed of the algorithms. For statistical purposes, the objective is to find the approximation model that best fits the data.

Sequential MARS has the flexibility to select any of the variants of MARS described in Section 3.2, depending on the needs of the application. It is preferable to use certain types of experimental designs that guarantee any partial set from the design is able to effectively represent the sample space. The diagram in Figure 4.1 represents the general approach for sequential MARS. The sequential methods for MARS are explained in the following sections. The methods are exemplified using a very simple coarse approximation for the initial iteration. This approximation is the same for all the methods. Table 4.1 displays the parameters used to generate the MARS approximation. The input data set considers 19 dimensions and it is not scaled $p = 0$. In general, the model is restricted up to three-way interactions; it employs ASR and only uses the forward coefficient restriction (FCR) Algorithm mentioned in Section 2.1.2, which follows the same structure as original Forward Algorithm (shown in appendix A) except for the one-dimensional transformation for the interaction terms previously described in Section 3.1.1. It is a nonconvex approximation, and it uses a piecewise-linear fit to determine the estimated model coefficients (β) (only requires the center knot).

Table 4.1. Input parameter settings for the initial MARS approximation.

M_{max}	10
Knots	5
Interactions	3
ASR	2
ASR difference	0.002
Robust	0
Robust tolerance	N/A
Original Backward Algorithm	0
Convex	0
Threshold	0.00
Smooth	0

The MARS approximation $\hat{y}^{(1)}$ generated is:

$\hat{y}^{(1)} = \beta_0 + \beta_1 \cdot B_1 + \beta_2 \cdot B_2 + \beta_3 \cdot B_3 + \beta_4 \cdot B_4$

where,

$\beta_0 = -0.6046, \quad \beta_1 = -0.7125, \quad \beta_2 = 49.7735, \quad \beta_3 = 0.2023, \quad \beta_4 = 1.2170.$

$B_1 = [-(x_6 - 0.9187)]_+,$

$B_2 = [+(x_6 - 0.9187)]_+,$

$B_3 = [-\{\frac{- \ \cdot \ 0.9187}{(- \ \cdot \ 0.9187-1)} + \frac{+ \ \cdot \ 0.7646}{(+ \ \cdot \ 0.7646-1)} + \frac{- \ \cdot \ x_6}{(1- \ - \ \cdot \ 0.9187)} + \frac{+ \ \cdot \ x_{19}}{(1- \ + \ \cdot \ 0.7646)}\}]_+,$

$B_4 = [+\{\frac{- \ \cdot \ 0.9187}{(- \ \cdot \ 0.9187-1)} + \frac{+ \ \cdot \ 0.7646}{(+ \ \cdot \ 0.7646-1)} + \frac{- \ \cdot \ x_6}{(1- \ - \ \cdot \ 0.9187)} + \frac{+ \ \cdot \ x_{19}}{(1- \ + \ \cdot \ 0.7646)}\}]_+.$

The MARS approximation is formed by four basis functions, a pair of univariate terms B_1 and B_2 and a pair of two-way interaction terms B_3 and B_4. The term β_0 represents the intercept, and the β_m terms represent the coefficients for their corresponding basis function m.

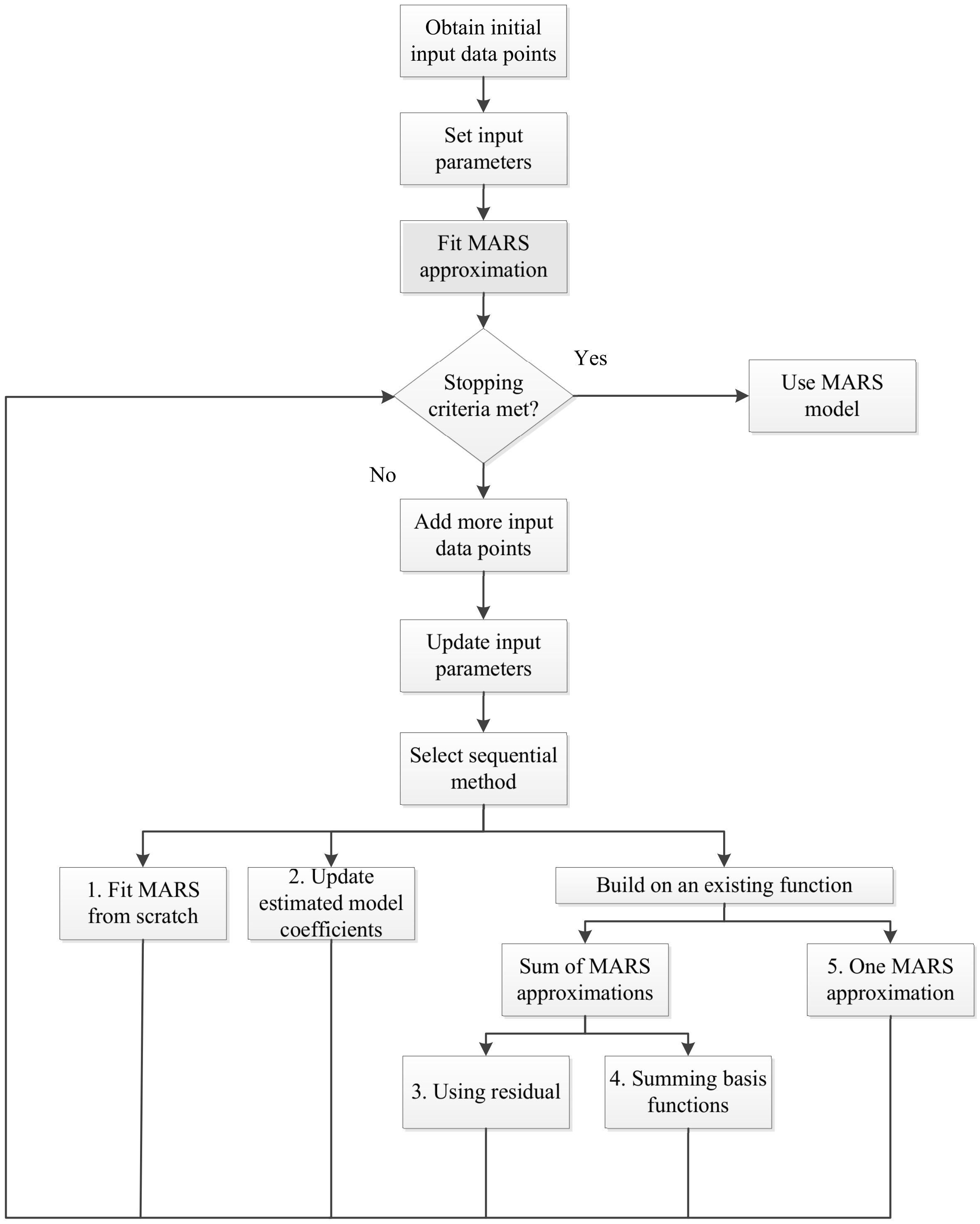

Obtain initial input data points
Set input parameters
Fit MARS approximation
Stopping criteria met?
Yes
Use MARS model
No
Add more input data points
Update input parameters
Select sequential method
1. Fit MARS from scratch
2. Update estimated model coefficients
Build on an existing function
Sum of MARS approximations
5. One MARS approximation
3. Using residual
4. Summing basis functions

Figure 4.1. Sequential MARS approach.

4.1 Sequential MARS 1 - Fit a MARS Function from Scratch.

Generating a new MARS approximation function from scratch is the simplest approach. It receives new points and fits the function according to the selected input parameters. The data points are being accumulated at each iteration and so, the input parameters are updated. The function structure changes every time that is, the basis functions with their corresponding parameters (coefficients β_m, variables $x_{v(l,m)}$, knots $k_{l,m}$ and directions $s_{l,m}$, ϕ_m) are different once the data set is updated. The increment in data points should be large enough to do a new fit. The approximation function is expected to be better by increasing the complexity of the MARS structure but still considering fewer data points.

Algorithm 4.1 describes the procedure. The input parameters for the MARS approximation for iteration 0 are described in Table 4.1, and the generated function is displayed at the beginning of this chapter. In this method, the MARS approximation from iteration 0 is not considered for the next iteration. Any new iteration will represent a completely new MARS approximation function. However, the input parameters of the previous iteration are considered for the next iteration. For example, for iteration 1, 10 more points are added. The updated input parameters are $N = 10 + 10 = 20$ points, the maximum number of basis functions also increments to $M_{max} = 5 + 5 = 10$ as well as the number of knots $T = 5 + 3$. The rest of the parameters remain the same. The Forward Coefficient Restriction (FCR) Algorithm is performed, and a piecewise-linear fit is use to generate a new MARS approximation. The new MARS approximation is:

$\hat{y}^{(2)} = \beta_0 + \beta_1 \cdot B_1 + \beta_2 \cdot B_2 + \beta_3 \cdot B_3 + \beta_4 \cdot B_4 \; \beta_5 \cdot B_5 + \beta_6 \cdot B_6 + \beta_7 \cdot B_7 + \beta_8 \cdot B_8 + \beta_9 \cdot B_9 + \beta_{10} \cdot B_{10}$

where,

$\beta_0 = -0.2976, \quad \beta_1 = -0.1005, \quad \beta_2 = 371.4221, \quad \beta_3 = 2.8405, \quad \beta_4 = 0.1282,$

$\beta_5 = -0.1134, \quad \beta_6 = 0.5699, \quad \beta_7 = -0.0758, \quad \beta_8 = 0.3332, \quad \beta_9 = -0.8178,$

$\beta_{10} = 0.0806,$

$B_1 = [-(x_{10} - 0.9895)]_+,$

$B_2 = [+(x_{10} - 0.9895)]_+,$

$B_3 = [-\{\frac{- \cdot 0.9895}{(- \cdot 0.9895-1)} + \frac{- \cdot 0.1385}{(- \cdot 0.1385-1)} + \frac{- \cdot x_{10}}{(1- - \cdot 0.9895)} + \frac{- \cdot x_9}{(1- - \cdot 0.1385)}\}]_+,$

$B_4 = [+\{\frac{- \cdot 0.9895}{(- \cdot 0.9895-1)} + \frac{- \cdot 0.1385}{(- \cdot 0.1385-1)} + \frac{- \cdot x_{10}}{(1- - \cdot 0.9895)} + \frac{- \cdot x_9}{(1- - \cdot 0.1385)}\}]_+,$

$B_5 = [-\{\frac{- \cdot 0.9895}{(- \cdot 0.9895-1)} + \frac{+ \cdot 0.6400}{(+ \cdot 0.6400-1)} + \frac{- \cdot x_{10}}{(1- - \cdot 0.9895)} + \frac{+ \cdot x_{18}}{(1- + \cdot 0.6400)}\}]_+,$

$B_6 = [+\{\frac{- \cdot 0.9895}{(- \cdot 0.9895-1)} + \frac{+ \cdot 0.6400}{(+ \cdot 0.6400-1)} + \frac{- \cdot x_{10}}{(1- - \cdot 0.9895)} + \frac{+ \cdot x_{18}}{(1- + \cdot 0.6400)}\}]_+,$

$B_7 = [-\{\frac{- \cdot 0.9895}{(- \cdot 0.9895-1)} + \frac{+ \cdot 0.1385}{(+ \cdot 0.1385-1)} + \frac{+ \cdot 0.4642}{(+ \cdot 0.4642-1)} + \frac{- \cdot x_{10}}{(1- - \cdot 0.9895)} + \frac{+ \cdot x_9}{(1- + \cdot 0.1385)} +$
$\frac{+ \cdot x_{11}}{(1- + \cdot 0.4642)}\}]_+,$

$B_8 = [+\{\frac{- \cdot 0.9895}{(- \cdot 0.9895-1)} + \frac{+ \cdot 0.1385}{(+ \cdot 0.1385-1)} + \frac{+ \cdot 0.4642}{(+ \cdot 0.4642-1)} + \frac{- \cdot x_{10}}{(1- - \cdot 0.9895)} + \frac{+ \cdot x_9}{(1- + \cdot 0.1385)} +$
$\frac{+ \cdot x_{11}}{(1- + \cdot 0.4642)}\}]_+,$

$B_9 = [-\{\frac{- \cdot 0.9895}{(- \cdot 0.9895-1)} + \frac{- \cdot 0.1385}{(- \cdot 0.1385-1)} + \frac{- \cdot 0.4555}{(- \cdot 0.4555-1)} + \frac{- \cdot x_{10}}{(1- - \cdot 0.9895)} + \frac{- \cdot x_9}{(1- - \cdot 0.1385)} +$
$\frac{- \cdot x_{16}}{(1- - \cdot 0.4555)}\}]_+,$

$B_{10} = [+\{\frac{- \cdot 0.9895}{(- \cdot 0.9895-1)} + \frac{- \cdot 0.1385}{(- \cdot 0.1385-1)} + \frac{- \cdot 0.4555}{(- \cdot 0.4555-1)} + \frac{- \cdot x_{10}}{(1- - \cdot 0.9895)} + \frac{- \cdot x_9}{(1- - \cdot 0.1385)} +$
$\frac{- \cdot x_{16}}{(1- - \cdot 0.4555)}\}]_+.$

The final approximation has a total of 10 basis functions. One pair of main effects B_1 and B_2, two pairs of two-way interaction effects B_3, B_4 and B_5, B_6, and two pairs of three-way interaction effects B_7, B_8 and B_9, B_{10}. Notice that the parent term for the three-way interaction terms B_7 and B_8 is basis function B_4. The second split in B_7 and B_8 has a positive sign while its parent term has a negative sign. This is because the three way interaction term is really using ϕ which is positive. Parameter ϕ is only required for the last split of the basis function. For the last split term $s = \phi$ (Section 3.1.1). Once the MARS approximation is generated, a testing data set is used to define the quality of fit based on the mean squared error (MSE).

This is a common approach for sequential designs; points are selected sequentially and the approximation is generated from scratch. However, the objective in this dissertation is to sequentially update the approximation. The points used for next iterations are not selected with any particular strategy, they are just the following consecutive points from a low-discrepancy sequence.

4.2 Sequential MARS 2 - Update the Estimated Model Coefficients.

In this sequential method, once an initial MARS function is determined, the β_m coefficients for each of the selected basis functions, as well as the intercept coefficient β_0, are updated as new data are added. The same structure is then kept for all the possible iterations required. That is the knots, the variables and the signs are maintained. The knots however, need to be re-standardized based on the updated data set. The least squares equation is refitted, so that the coefficients can be updated, as well as the lack of fit calculation to evaluate the performance of the updated MARS approximation function. Algorithm 4.2 describes the procedure.

The input parameters for the MARS approximation for iteration 0 are described in Table 4.1 and the generated function is displayed at the beginning of this chapter. For iteration 1, 10 more points are added, $N = 10 + 10 = 20$. Since this method only refits the β_m coefficients for next iterations, there is no need to update the maximum number of basis functions (M_{max}) or the total number of eligible knots (T). The current function structure (basis functions) remains unchanged. Once the algorithm reads the total points, the data matrix x ($N \ x \ n$ input variables) is standardized for numerical stability based on the midrange and half range from $N = 20$. The vector of response variables is read and $\bar{y}$ is computed. The knots from the existing MARS approximation (iteration 0) are based on the midrange and half range from $N = 10$, therefore, they need to be re-standardized based on the total number of points. In $\hat{y}^{(1)}$

Algorithm 4.1 Sequential MARS 1 Algorithm

```
MARS approximation for iteration $i = 0$ has been previously created.
for all $i = 1, ..., i = iter$ iterations do
  Initialize $B_1(x) = 1$, $M = 1$, $LOF^* = \infty$.
  while $M < M_{max}$ do
    for all existing basis functions $B_m$, $m = 0, ..., M - 1$ do
      if interaction order $l_m < L_m$ then
        for all eligible covariates $x_v$ $(v \in 1, ..., n)$ do
          for all eligible knots $k$ ($k \in$ knot set for $x_v$ s.t. $B_m(k) > 0$) do
            if new pair is formed by univariate terms regress $y$ on:
            $\sum_{i=1}^{M-1} B_i(x) + B_m \cdot [-(x-k)]_+ + B_{m+1} \cdot [+(x-k)]_+$;
            if new pair is formed by interaction terms regress $y$ on:
            $\sum_{i=1}^{M-1} B_i(x) + B_m \cdot [-\phi_m \cdot z_m]_+ + B_{m+1} \cdot [+\phi_m \cdot z_m]_+$;
            if convex, check restrictions on estimated model coefficients.
            calculate lack-of-fit ($LOF$)
            if $LOF < LOF^*$ then
              $LOF^* = LOF, \;\; m^* = m, \;\; v^* = v, \;\; k* = k.$
            end if
          end for $k$
        end for $v$
      end if
    end for $m$
    Add and orthonormalize basis functions $B_M(x)$ and $B_{M+1}(x)$, $(m^*, x_{v^*}, k^*)$,
    $M = M + 2$.
  end while $M$
  perform backward routine if required by the input parameters.
  compute mean squared error (MSE) on testing data set.
end for $i$
```

the only variables selected were x_6 and x_{19}. The midrange based on $N = 10$ points for variable x_6 was $23,492.624850$ and the half range was $23,380.532050$; similarly for variable x_{19}, the midrange was 0.059995 and the half range was 0.049995. Based on $N = 20$, for variable x_6, the new midrange is $26,736.813507$ and the new half range is $26,624.720707$. For variable x_{19}, the new midrange is 0.062360 and the new half range is 0.052360. To obtain the actual value for the knots:

$$\text{knot} = (\text{knot} \cdot \text{half range}) + \text{midrange};$$

and to re-standardize based on all the points:

$$\text{knot} = (\text{knot} - \text{new midrange})/\text{new half range}.$$

The actual knot value for variable x_6 is:

$$\text{knot} = (0.9187 \cdot 23,380.532050) + 23,492.624850 = 44,972.31964$$

and the re-standardize knot value is:

$$\text{knot} = (44,972.31964 - 26,736.813507)/26,624.720707 = 0.68490.$$

The actual knot value for variable x_{19} is:

$$\text{knot} = (0.7646 \cdot 0.049995) + 0.059995 = 0.09822$$

and the re-standardize knot value is:

$$\text{knot} = (0.09822 - 0.062360)/0.052360 = 0.68489.$$

Note that the new standardized value for both variables turned out to be very similar.

The coefficients are updated based on the linear least-squares fit procedure as in original MARS [18]. One popular technique for numerically performing this fit is based on the QR decomposition ($A = QR$, where Q is an orthogonal matrix and R is an upper triangular matrix) however, Friedman prefers the Cholesky decomposition ($A = LL^*$, where L is a lower triangular matrix with real and positive diagonal entries and L^* denotes the conjugate transpose of L) because even though the QR decomposition has superior numerical properties, the Cholesky decomposition is faster to compute.

Given the Nx$(M+1)$ data matrix B, where $B_{ij} = B_j(x_i)$ the following equation needs to be solved for the vector of basis coefficients a:

$$B^T Ba = B^T y \tag{4.1}$$

where y represents the length N vector of the response values. If the basis functions are centered, Equation 4.1 can be written as:

$$Va = c \tag{4.2}$$

where

$$V_{ij} = \sum_{k=1}^{N} B_j(x_k)[B_i(x_k) - \bar{B}_i], \quad \text{and} \tag{4.3}$$

$$c_i = \sum_{k=1}^{N} (y_k - \bar{y})B_i(x_k), \tag{4.4}$$

where $\bar{B}$ and $\bar{y}$ represents the corresponding averages over the data. The intercept coefficient β_0 is computed as follows:

$$a_0 = \bar{y} - \sum_{i=1}^{M} a_i \bar{B}_i. \tag{4.5}$$

Continuing with the existing structure from iteration 0, and following the above equations, the updated coefficients for iteration 1 are:

$\beta_0 = 0.6698, \quad \beta_1 = -0.7543, \quad \beta_2 = -0.7228, \quad \beta_3 = -0.0257, \quad \beta_4 = -0.0522.$

Once the MARS approximation is updated, a testing data set is used to define the quality of fit based on the mean squared error (MSE).

This sequential approach is the one that requires the least computational effort since it does not increase the complexity of the structure at each iteration. The purpose of refitting the estimated model coefficients is to obtain a more accurate approximation by considering more data points. The quality of the fit might not be as good as other approaches but it shows stability with fewer data points. Having a simple structure might be advantageous for certain applications. For optimization purposes, this might be the easiest to optimize.

4.3 Sequential MARS 3 - Build on an Existing MARS Function: Sum of MARS Approximations Based on Residuals.

This approach conserves the initial MARS structure and updates the overall fit based on new data. Once new input data points are added, the existing MARS function is used to calculate the residuals $e = y - \hat{y}$ for all the points. The residuals then are used as the response variable for the next iteration. The updated estimated value is the sum of the previous iteration estimated value plus the estimated value from the updated MARS function, $\hat{y}_{new} = \hat{y}^{(1)} + \hat{y}^{(2)}$. In theory, this should provide a more robust approximation since the method is trying to compensate the remaining unexplained variability from previous iterations by updating the function using the residuals as the response variable for new iterations. Algorithm 4.3 describes the procedure. The following example demonstrates the method.

The input parameters for the MARS approximation for iteration 0 are described in Table 4.1 and the generated function is displayed at the beginning of this chapter. For iteration 1, 10 more points are added. The new parameters are $N = 10 + 10 = 20$, $M_{max} = 5$ and $T = 3$. Once the algorithm reads the total points, it determines the set of eligible knots. The data matrix x is standardized for numerical stability based on the midrange and half range from $N = 20$. The vector of response variables is read and $\bar{y}$ is computed. The knots from the existing MARS approximation (iteration 0) are based on the midrange and half

Algorithm 4.2 Sequential MARS 2 Algorithm

```
MARS approximation for iteration $i = 0$ has been previously created.
for all $i = 1, ..., iter$ iterations do
    for all existing basis functions $B_{m^*}$, $m^* = 1, ..., M^*$ do
        for all selected covariates $x_{v^*}$ do
            for all selected knots $k^*$ do
                obtain actual knot value: $k = (k \cdot \text{half range}) + \text{midrange}$,
                re-standardize based on new $N$ data points:
                $k = (k - \text{new midrange})/\text{new half range}$.
            end for $k$
        end for $x_{v^*}$
    end for $B_{m^*}$
    for all basis functions $B_{m^*}$ do
        if piecewise-linear fit then
            follow Equation: 2.3
        else if quintic fit then
            follow Equations: 3.6 and 3.7
        end if
    end for $B_{m^*}$
    for all $\beta_{m^*}$ coefficients $m^* = 1, ..., M^*$ do
        solve $V_{ij} = \sum_{k=1}^{N} B_j(x_k)[B_i(x_k) - \bar{B}_i]$, and $c_i = \sum_{k=1}^{N}(y_k - \bar{y})B_i(x_k)$.
    end for $\beta_{m^*}$
    for intercept coefficient $\beta_0$ solve $\bar{y} - \sum_{i=1}^{M} a_i \bar{B}_i$.
    calculate lack-of-fit ($LOF$).
    compute mean squared error (MSE) on testing data set.
end for $i$
```

range from $N = 10$, therefore, they need to be re-standardize based on the total number of points. Same as before, to obtain the actual value for the knots:

$$\text{knot} = (\text{knot} \cdot \text{half range}) + \text{midrange};$$

and to re-standardize based on all the points:

$$\text{knot} = (\text{knot} - \text{new midrange})/\text{new half range}.$$

The β_m coefficients need to be updated following the same procedure as in Section 4.2. The updated exiting MARS approximation for iteration 0 is:

$\hat{y}^{(1)} = \beta_0 + \beta_1 \cdot B_1 + \beta_2 \cdot B_2 + \beta_3 \cdot B_3 + \beta_4 \cdot B_4$

where,

$\beta_0 = 0.6698, \quad \beta_1 = -0.7543, \quad \beta_2 = -0.7228, \quad \beta_3 = -0.0257, \quad \beta_4 = -0.0522,$

$B_1 = [-(x_6 - 0.6849)]_+,$

$B_2 = [+(x_6 - 0.6849)]_+,$

$B_3 = [-\{\frac{- \,\cdot\, 0.6849}{(- \,\cdot\, 0.6849-1)} + \frac{+ \,\cdot\, 0.6849}{(+ \,\cdot\, 0.6849-1)} + \frac{- \,\cdot\, x_6}{(1- \; - \,\cdot\, 0.6849)} + \frac{+ \,\cdot\, x_{19}}{(1- \; + \,\cdot\, 0.6849)}\}]_+$, and

$B_4 = [+\{\frac{- \,\cdot\, 0.6849}{(- \,\cdot\, 0.6849-1)} + \frac{+ \,\cdot\, 0.6849}{(+ \,\cdot\, 0.6849-1)} + \frac{- \,\cdot\, x_6}{(1- \; - \,\cdot\, 0.6849)} + \frac{+ \,\cdot\, x_{19}}{(1- \; + \,\cdot\, 0.6849)}\}]_+.$

In order to generate the next MARS approximation function, the residuals should be calculated. The vector of y responses ($N = 20$) needs to be evaluated using the updated MARS approximation. This evaluation can use a piecewise-linear fit or a smoothing routine according to the initial set of parameters to generate the function. This example used a piecewise-linear fit. Using the information from the first input point, the residual is calculated as follows:

$$y_1 = -0.63537 \;\; \hat{y}_1^{(1)} = -0.71274,$$

$$e_1 = -0.63537 - (-0.71274) = 0.07737.$$

The same calculation is made for the total set of points. The residuals are then used as a response y to generate the approximation for iteration 1. The new MARS approximation is: $\hat{y}^{(2)} = \beta_0 + \beta_1 \cdot B_1 + \beta_2 \cdot B_2 + \beta_3 \cdot B_3 + \beta_4 \cdot B_4$

where,

$\beta_0 = -0.7193, \quad \beta_1 = -0.1208, \quad \beta_2 = 4.4431, \quad \beta_3 = 0.0971, \quad \beta_4 = 3.3969,$

$B_1 = [-(x_{17} - 0.5834)]_+,$

$B_2 = [+(x_{17} - 0.5834)]_+,$

$B_3 = [-\{\frac{+\ \cdot\ 0.5834}{(+\ \cdot\ 0.5834-1)} + \frac{+\ \cdot\ 0.7563}{(+\ \cdot\ 0.7563-1)} + \frac{+\ \cdot\ x_{17}}{(1-\ +\ \cdot\ 0.5834)} + \frac{+\ \cdot\ x_6}{(1-\ +\ \cdot\ 0.7563)}\}]_+$, and

$B_4 = [+\{\frac{+\ \cdot\ 0.5834}{(+\ \cdot\ 0.5834-1)} + \frac{+\ \cdot\ 0.7563}{(+\ \cdot\ 0.7563-1)} + \frac{+\ \cdot\ x_{17}}{(1-\ +\ \cdot\ 0.5834)} + \frac{+\ \cdot\ x_6}{(1-\ +\ \cdot\ 0.7563)}\}]_+.$

The MARS approximation is formed by a pair of main effects (B_1 and B_2) and a pair of two-way interaction effects (B_3 and B_4). There is a total of four basis functions and only variables x_6 and x_{17} were selected. To evaluate the testing data set, the MARS approximation for all iterations should be used:

$$\hat{y}_1^{(1)} = -0.65487 \quad \hat{y}_1^{(2)} = 0.07923,$$

$$\hat{y}_{new} = -0.65487 + 0.07923 = -0.57564.$$

Mean squared error (MSE) is used to define the quality of fit. This method may be recommended to use when the underlying function does not have a significant number of interaction effects.

4.4 Sequential MARS 4 - Build on an Existing MARS Function: Sum of MARS Approximations.

This sequential method is based on the same concept as sequential MARS 3. The new approximation is build on existing basis functions following a boosting principle except that it does not use the residuals to estimate the next MARS approximation. The structure of the approximation is based on adding new basis functions at each iteration. Algorithm 4.4 describes the procedure. The following example demonstrates the method.

Algorithm 4.3 Sequential MARS 3 Algorithm

```
MARS approximation for iteration $i = 0$ has been previously created.
**for all** $i = 1, ..., iter$ iterations **do**
    follow Algorithm 4.2 to re-standardize selected knots $k^*$ and
    to update $\beta_{m^*}$ coefficients.
    **for all** training data points $y = 1, ..., N$ **do**
        evaluate updated function $\hat{y}^{(i)}(x) = \beta_0 + \sum_{m=1}^{M} \beta_m x_m$, and
        calculate residuals $e = y - \hat{y}^{(i)}$.
    **end for** $y$
    follow Algorithm 4.1 to generate new MARS approximation function using resid-
    uals as the response $e = y$,
    evaluate testing data set using MARS approximations from $i - 1, ..., iter$,
    add new and existing MARS approximations $\hat{y}_{new} = \hat{y}^{(1)} + \hat{y}^{(2)} +, , \hat{y}^{(iter+1)}$.
    compute mean squared error (MSE) on testing data set.
**end for** $i$
```

The input parameters for the MARS approximation for iteration 0 are described in Table 4.1 and the generated function is displayed at the beginning of this chapter. For iteration 1, 10 more points are added. The updated input parameters are $N = 10 + 10 = 20$, $M_{max} = 5$ and $T = 3$. Once the algorithm reads the total points, it determines the set of eligible knots. For this sequential approach, the initial knots (from iteration 0) are fixed for the rest of the iterations, allowing the ability to only use the initial set of eligible knots or to add more knots in the next iterations.

The MARS procedure to select the knots is represented in Algorithm 4.5. Note: this method only applies for setting central knots that is, using a piecewise-linear fit. If one requires smoothing the function, side knots are needed and they are set

Algorithm 4.4 Sequential MARS 4 Algorithm

```
MARS approximation for iteration $i = 0$ has been previously created.
**for all** $i = 1, ..., iter$ iterations **do**
    define the set of new eligible knots ($T_i$),
    save the set of total eligible knots (total_knots $= T_i + T_{i-1}$) for next iteration $[i+1]$.
    follow Algorithm 4.2 to re-standardize selected knots $k^*$
    save updated MARS approximation $\hat{y}^{(i-1)}(x) = \beta_0 + \sum_{m=1}^{M} \beta_m x_m$.
    follow Algorithm 4.1 to generate new MARS approximation function based on new $N$ data points, new $M_{max}$ and total_knots,
    save new MARS approximation $\hat{y}^{(i)}(x) = \beta_0 + \sum_{m=1}^{M} \beta_m x_m$.
    add new and existing MARS approximations $\hat{y}_{new} = \hat{y}^{(1)} + \hat{y}^{(2)} +, , \hat{y}^{(iter+1)}$.
    follow Algorithm 4.2 to update $\beta_{m^*}$ coefficients.
    compute mean squared error (MSE) on testing data set.
**end for** $i$
```

differently. Therefore this sequential approach is limited to use in the MARS variants that require a piecewise-linear fit only.

The procedure is illustrated in the following example. Assume there are $N = 10$ initial input points for each variable i and a total of variables $n = 3$. The input points are not required to be in any particular order (Table 4.2), and the number of desired knots is $T_0 = T = 5$.

The algorithm sorts the points in ascending order (xval) and counts the number of levels (p) for each variable i. If the minimum number of levels for any variable is fewer than the desired number of knots, then T is set $p - 2$ knots. The sorted points are shown in Table 4.3. Note that for variable $i = 1$ the input points coincide to be in

Table 4.2. Initial input points for variables $i = 1, ..., n$, $N = 10$.

Index	$i = 1$	Index	$i = 2$	Index	$i = 3$
1	92.17	1	145.61	1	1274.60
2	2,966.61	2	45,557.34	2	200,350.61
3	5,841.06	3	31,970.15	3	398,790.60
4	8,738.50	4	59,217.19	4	597,866.60
5	11,589.95	5	63,722.03	5	279,853.81
6	14,487.39	6	18,310.30	6	80,777.80
7	17,338.85	7	41,052.50	7	518,363.41
8	20,236.29	8	13,805.46	8	319,287.40
9	23,087.74	9	68,299.54	9	557,797.00
10	25,985.18	10	22,887.80	10	438,860.21

ascending order. Additionally, all variables have the same number of levels ($p = 10$), but this is not always the case. Following Algorithm 4.5, the index for the selected knots for variable $i = 1$ is defined as follows:

$$\text{knotspace} = (10 - 1)/5 = 1.8,$$

$$\text{firstknot} = 1.8/2 = 0.9,$$

$$\text{knotidx} = 0.9, \quad \text{knot}[1][1] = \text{xval}[1][0.9 + 0.5].\text{idx} = \text{xval}[1][2].\text{idx},$$

knotidx increments at each iteration:

$$\text{knotidx} = 0.9 + 1.8, \quad \text{knot}[1][2] = \text{xval}[1][2.7 + 0.5] = \text{xval}[1][4]$$

$$\text{knotidx} = 2.7 + 1.8, \quad \text{knot}[1][3] = \text{xval}[1][4.5 + 0.5] = \text{xval}[1][6]$$

$$\text{knotidx} = 4.5 + 1.8, \quad \text{knot}[1][4] = \text{xval}[1][6.3 + 0.5] = \text{xval}[1][7]$$

$$\text{knotidx} = 6.3 + 1.8, \quad \text{knot}[1][5] = \text{xval}[1][8.1 + 0.5] = \text{xval}[1][9].$$

For this example the index for the selected knots for the three variables is the same since the number of levels is the same and the number of knots is constant across variables. Also, note that the algorithm never selects the extreme points (minimum and maximum). In this case, the distribution of knots is uneven since knot t_3 is adjacent to knot t_4. If $p[i] \mod 2 = 0$ and $T \mod 2 > 0$ the knots are asymmetrically distributed over the number of levels of the variable.

Table 4.3. Selected knots for variables $i = 1, ..., n$.

	Order for knot	Index	$i = 1$	Index	$i = 2$	Index	$i = 3$
	1	1	92.17	1	145.61	1	1,274.60
$t_1 =$	2	2	2,966.61	8	13,805.46	6	80,777.80
	3	3	5,841.06	6	18,310.30	2	200,350.61
$t_2 =$	4	4	8,738.50	10	22,887.80	5	279,853.81
	5	5	11,589.95	3	31,970.15	8	319,287.40
$t_3 =$	6	6	14,487.39	7	41,052.50	3	398,790.60
$t_4 =$	7	7	17,338.85	2	45,557.34	10	438,860.21
	8	8	20,236.29	4	59,217.19	7	518,363.41
$t_5 =$	9	9	23,087.74	5	63,722.03	9	557,797.00
	10	10	25,985.18	9	68,299.54	4	597,866.61

To add more knots, the following procedure is performed. Continuing with the example, 10 new points are added for the new MARS approximation, $N = 20$ and 3 additional knots ($T_1 = 3$) are required to be added; the total number of knots is $T = T_0 + T_1 = 8$. The number of levels p for each variable i is counted. The number of levels for variable $i = 1$ is $p = 18$, while the number of levels for variables $i = 2$ and $i = 3$ is $p = 20$. The median is calculated with the purpose to determine how many existing knots are located above the median and below the median. The medians for each variable are: median[1]=18,787.57, median[2]=34,258.90 and median [3]=299,570.61. Table 4.4 shows the new points already sorted with their corresponding index. There is a horizontal line for each vector that represents the break point according to the median. Average and midpoint were also considered to use, behaving similarly as the median when the data are normally distributed however, when the data are skewed the breakpoint cannot be even. By using the median, a balanced number of knots

Algorithm 4.5 Algorithm for setting knots

```
for all $i = 1, ..., n$ variables do
    knotspace = $p[i] - 1/T$;
    firstknot = knotspace/2;
    knotidx = firstknot;
    for all $t = 1, ..., T$ knots do
        knot[$i$][$t$] = xval[$i$][knotidx + 0.5];
        Note: [knotidx + 0.5] is rounded up to the nearest integer.
        knotidx$_+$ = knotspace;
    end for
end for
```

above and below the break point is guaranteed. For variable $i = 1$, the number of knots previously selected that are above the median is $\text{part}_1 = 4$ and below $\text{part}_2 = 1$; similarly for variable $i = 2$, $\text{part}_1 = 2$ and $\text{part}_2 = 3$ and for variable $i = 3$, $\text{part}_1 = 2$ and $\text{part}_2 = 3$. Algorithm 4.6 determines how many new knots need to be added above (t_1) and below (t_2) the median. For variable $i = 1$, $t_1 = 0$ and $t_2 = 3$, for variable $i = 2$, $t_1 = 2$ and $t_2 = 1$ and for variable $i = 3$, $t_1 = 2$ and $t_2 = 1$. Then the existing knots are temporarily removed from the vector of points for each variable in order to select the new candidates for knots and avoid overlapping with existing ones. In the same manner as before, the algorithm sorts the points in ascending order and counts the number of levels p for each variable i. The number of levels for variable $i = 1$ after removing existing knots is $p = 13$, while the number of levels for variables $i = 2$ and $i = 3$ is $p = 15$. Also, if the minimum number of levels for any variable is fewer than the desired number of knots, then T_1 is set $p - 2$ knots. The next step is to divide p for each variable, $\text{half\_space}_1 = p/2$ and $\text{half\_space}_2 = p - \text{half\_space}_1$.

Note: half_space$_1$ is rounded down to the nearest integer. Now, the same procedure described in Algorithm 4.5 is performed for half_space$_1$ and half_space$_2$, except that for half_space$_2$, knotidx has to be initialized as knotidx = half_space$_1$. There should be a need to assign new knots in order to perform Algorithm 4.5 that is, the number of new knots required above the median needs to be greater than zero $t_1 > 0$ for half_space$_1$ and similarly the number of new knots required below the median needs to be greater than zero $t_2 > 0$ for half_space$_2$. Table 4.4 shows the existing knots (which are the same as in Table 4.3) highlighted in yellow and the preliminary new knots highlighted in blue. For variable $i = 1$, the distribution of the existing and new knots seems to be the optimal since the new knots have not adjacency with existing knots. Nevertheless, this is not the case for variables $i = 2$ and $i = 3$. Consequently, an additional routine needs to be performed.

Once the new preliminary knots have been defined, the vector of points for each variable is classified. That is, if point$[i][j]$ ($i = 1, ...i = n$ and $j = 1, ..., j = p$) is equal to existing knot then point$[i][j] = 1$; if point$[i][j]$ is equal to new preliminary knot then point$[i][j] = 2$; if point$[i][j]$ is not equal to any existing or new preliminary knot then point$[i][j] = 0$; Algorithm 4.7 counts the space (number of points) between existing knots (point$[i][j] = 1$) and reassigns the new preliminary knots. Essentially, the same logic for assigning knots described in Algorithm 4.5 is now performed for each space. It also identifies the final eligible knot as point$[i][j] = 1$, so at the end, any point with a value of 1 is a selected candidate knot. Table 4.5 displays the final set of knots, the values highlighted in magenta color represent the knots that needed to be relocated from the preliminary selection (Table 4.4). Now the knots are better distributed along the vector of points. Adjacency within the initial selected knots (yellow) cannot be avoided since the purpose of the algorithm is to maintain the initial set for future iterations.

Algorithm 4.6 Algorithm for assigning number of new knots.

```
Initialize $T = 8$, $T_1 = 3$, $\text{half}_1 = T/2 = 4$, $\text{half}_2 = T - \text{half}_1 = 4$.
Note: $\text{half}_1$ is rounded down to the nearest integer.
for all $i = 1, ..., n$ variables do
  if $\text{part}_1[i] < \text{half}_1$ then
    $t_1[i] = \text{half}_1 - \text{part}_1[i]$;
    if $t_1[i] > T_1$ then
      $t_1[i] = T_1$;
    else if  then
      $t_1[i] = 0$;
    end if
  end if
  if $\text{part}_2[i] < \text{half}_2$ then
    $t_2[i] = \text{half}_2 - \text{part}_2[i]$;
    if $t_2[i] > T_1$ then
      $t_2[i] = T_1$;
    else if  then
      $t_2[i] = 0$;
    end if
  end if
end for
```

Algorithm 4.7 Algorithm to determine the final set of knots.

```
Initialize space=0, knots=0.
for all $i = 1, ..., n$ variables do
    number = 1;
    for all $j = 1, ..., p[i]$ number of levels do
        if point[$i$][$j$] = 1 then
            $m = j$, space = $m$ − number.
            if space > 0 then
                for all $g$ = number − 1, ..., $m$ points do
                    if point[$i$][$g$] = 2 then
                        knots$_+$ = 1;
                        point[$i$][$g$] = 0;
                    end if
                end for
            end if
            if knots > 0 then
                relocate new knots following Algorithm 4.5
            end if
            knots= 0;
            number = $m + 1$;
        end if
    end for
    space =0;
    knots =0;
end for
```

Table 4.4. Existing knots (yellow) and preliminary new selected knots (blue) for variables $i = 1, ..., n$, $N = 20$.

Order for knot	Index	$i = 1$	Order for knot	Index	$i = 2$	Order for knot	Index	$i = 3$
1	1	92.17	1	1	145.61	1	1	1,274.60
2	17	101.36	2	17	232.80	2	17	2,801.06
3	2	2,966.61	3	13	4,686.78	3	12	41,026.20
4	18	2,975.81	4	12	9,227.96	4	6	80,777.80
5	3	5,841.06	5	8	13,805.46	5	15	120,847.41
6	4	8,738.50	6	6	18,310.30	6	11	160,281.00
7	5	11,589.95	7	10	22,887.80	7	2	200,350.61
8	6	14,487.39	8	15	27,392.65	8	18	201,622.67
9	7	17,388.85	9	3	31,970.15	9	16	239,784.20
10	8	20,236.29	10	19	32,042.81	10	5	279,853.81
11	9	23,087.74	11	11	36,474.99	11	8	319,287.40
12	10	25,985.18	12	7	41,052.50	12	13	359,357.01
13	11	28,836.63	13	2	45,557.34	13	3	398,790.60
14	12	31,734.07	14	18	45,630.00	14	19	400,062.65
15	13	34,585.52	15	14	50,134.84	15	10	438,860.21
16	14	37,482.97	16	16	54,639.69	16	14	478,293.80
17	15	40,334.42	17	4	59,217.19	17	7	518,363.41
18	16	43,231.86	18	20	59,289.85	18	9	557,797.00
			19	5	63,722.03	19	4	597,866.61
			20	9	68,299.54	20	20	599,138.66

After setting the knots, the data matrix x is standardized for numerical stability based on the midrange and half range from $N = 20$. The vector of response variables is read and $\bar{y}$ is computed. The knots from the existing MARS approximation (iteration 0) are based on the midrange and half range from $N = 10$, therefore, they need to be re-standardize based on the total number of points. As before, to obtain the actual value for the knots, calculate:

$$\text{knot} = (\text{knot} \cdot \text{half\ \ range}) + \text{midrange};$$

and re-standardize based on all the points:

$$\text{knot} = (\text{knot} - \text{new\ \ midrange})/\text{new\ \ half\ \ range}.$$

The β_m coefficients do not need to be updated at this point since, they will be defined

Table 4.5. Final knots selection. Existing knots (yellow), preliminary new selected knots (blue) and knots that needed to be relocated (magenta) for variables $i = 1, ..., n$, $N = 20$.

Order for knot	Index	$i = 1$	Order for knot	Index	$i = 2$	Order for knot	Index	$i = 3$
1	1	92.17	1	1	145.61	1	1	1,274.60
2	17	101.36	2	17	232.80	2	17	2,801.06
3	2	2,966.61	3	13	4,686.78	3	12	41,026.20
4	18	2,975.81	4	12	9,227.96	4	6	80,777.80
5	3	5,841.06	5	8	13,805.46	5	15	120,847.41
6	4	8,738.50	6	6	18,310.30	6	11	160,281.00
7	5	11,589.95	7	10	22,887.80	7	2	200,350.61
8	6	14,487.39	8	15	27,392.65	8	18	201,622.67
9	7	17,388.85	9	3	31,970.15	9	16	239,784.20
10	8	20,236.29	10	19	32,042.81	10	5	279,853.81
11	9	23,087.74	11	11	36,474.99	11	8	319,287.40
12	10	25,985.18	12	7	41,052.50	12	13	359,357.01
13	11	28,836.63	13	2	45,557.34	13	3	398,790.60
14	12	31,734.07	14	18	45,630.00	14	19	400,062.65
15	13	34,585.52	15	14	50,134.84	15	10	438,860.21
16	14	37,482.97	16	16	54,639.69	16	14	478,293.80
17	15	40,334.42	17	4	59,217.19	17	7	518,363.41
18	16	43,231.86	18	20	59,289.85	18	9	557,797.00
			19	5	63,722.03	19	4	597,866.61
			20	9	68,299.54	20	20	599,138.66

based on the complete new MARS approximation that is, including the existing (four) and new basis functions. The new approximation ($\hat{y}^{(2)}$) is now generated. The following is the MARS approximation function: $\hat{y}^{(2)} = \beta_0 + \beta_1 \cdot B_1 + \beta_2 \cdot B_2 + \beta_3 \cdot B_3 + \beta_4 \cdot B_4$ where,

$\beta_0 = 0.1288, \quad \beta_1 = 0.8992, \quad \beta_2 = 2.7123, \quad \beta_3 = 14.0846, \quad \beta_4 = -1.0776,$

$B_1 = [-(x_{17} - 0.4819)]_+,$

$B_2 = [+(x_{17} - 0.4819)]_+,$

$B_3 = [-\{\frac{- \cdot 0.4819}{(- \cdot 0.4819-1)} + \frac{- \cdot 0.9875}{(- \cdot 0.9875-1)} + \frac{- \cdot x_{17}}{(1- \; - \cdot 0.9875)} + \frac{- \cdot x_{18}}{(1- \; - \cdot 0.9875)}\}]_+$, and

$B_4 = [+\{\frac{- \ \cdot \ 0.4819}{(- \ \cdot \ 0.4819-1)} + \frac{- \ \cdot \ 0.9875}{(- \ \cdot \ 0.9875-1)} + \frac{- \ \cdot \ x_{17}}{(1- \ - \ \cdot \ 0.9875)} + \frac{- \ \cdot \ x_{18}}{(1- \ - \ \cdot \ 0.9875)}\}]_+.$

The new MARS approximation is formed by four basis functions, a pair of univariate terms B_1 and B_2 and a pair of 2-way interaction terms B_3 and B_4. The term β_0 represents the intercept term and the β_m terms represent the coefficients for their corresponding basis function m. Now $\hat{y}^{(1)}$ and $\hat{y}^{(2)}$ are added together. The index for the new MARS approximation starts from the last number of basis function from existing MARS approximation plus one.

$\hat{y} = \hat{y}^{(1)} + \hat{y}^{(2)} = \beta_0 + \beta_1 \cdot B_1 + \beta_2 \cdot B_2 + \beta_3 \cdot B_3 + \beta_4 \cdot B_4 + \beta_5 \cdot B_5 + \beta_6 \cdot B_6 + \beta_7 \cdot B_7 + \beta_8 \cdot B_8$

where,

$\beta_0 = -0.6046$ (from $\hat{y}^{(1)}$) + $\beta_0 = 0.1288$ (from $\hat{y}^{(2)}$),

$\beta_1 = -0.7125, \quad , \beta_2 = 49.7735, \quad \beta_3 = 0.2023, \quad \beta_4 = 1.2170,$

$\beta_5 = 0.8992, \quad \beta_6 = 2.7123, \quad \beta_7 = 14.0846, \quad \beta_8 = -1.0776,$

$B_1 = [-(x_6 - 0.6849)]_+,$

$B_2 = [+(x_6 - 0.6849)]_+,$

$B_3 = [-\{\frac{- \ \cdot \ 0.6849}{(- \ \cdot \ 0.6849-1)} + \frac{+ \ \cdot \ 0.6849}{(+ \ \cdot \ 0.6849-1)} + \frac{- \ \cdot \ x_6}{(1- \ - \ \cdot \ 0.6849)} + \frac{+ \ \cdot \ x_{19}}{(1- \ + \ \cdot \ 0.6849)}\}]_+,$

$B_4 = [+\{\frac{- \ \cdot \ 0.6849}{(- \ \cdot \ 0.6849-1)} + \frac{+ \ \cdot \ 0.6849}{(+ \ \cdot \ 0.6849-1)} + \frac{-1 \ \cdot \ x_6}{(1- \ - \ \cdot \ 0.6849)} + \frac{+ \ \cdot \ x_{19}}{(1- \ + \ \cdot \ 0.6849)}\}]_+,$

$B_5 = [-(x_{17} - 0.4819)]_+,$

$B_6 = [+(x_{17} - 0.4819)]_+,$

$B_7 = [-\{\frac{- \ \cdot \ 0.4819}{(- \ \cdot \ 0.4819-1)} + \frac{- \ \cdot \ 0.9875}{(- \ \cdot \ 0.9875-1)} + \frac{- \ \cdot \ x_{17}}{(1- \ - \ \cdot \ 0.9875)} + \frac{- \ \cdot \ x_{18}}{(1- \ - \ \cdot \ 0.9875)}\}]_+,$

$B_8 = [+\{\frac{- \ \cdot \ 0.4819}{(- \ \cdot \ 0.4819-1)} + \frac{- \ \cdot \ 0.9875}{(- \ \cdot \ 0.9875-1)} + \frac{- \ \cdot \ x_{17}}{(1- \ - \ \cdot \ 0.9875)} + \frac{- \ \cdot \ x_{18}}{(1- \ - \ \cdot \ 0.9875)}\}]_+.$

The β_m coefficients are now updated following the same procedure as in Section 4.2. The updated coefficients are:

$\beta_0 = -0.0328, \quad \beta_1 = 0.0031, \quad \beta_2 = 0.5986, \quad \beta_3 = 0.0739, \quad \beta_4 = 0.2367,$

$\beta_5 = 1.0545, \quad \beta_6 = 2.6505, \quad \beta_7 = 14.4165, \quad \beta_8 = -1.2453.$

The final approximation has four main effects and four interaction effects. The refit of coefficients was based on a piecewise-linear fit. After finishing the MARS approximation, a testing data set is used to define the quality of fit based on the mean squared error (MSE). As sequential MARS 3, this method may be recommended to use when the underlying function does not have a significant number of interaction effects.

4.5 Sequential MARS 5 - Build on an Existing MARS Function: One MARS Approximation.

This sequential approach allows adding more basis functions as in the sequential MARS 4 method (Section 4.4). The difference is that this method allows forming interaction terms on existing basis functions. That is, the new input data points can generate main effects, new interaction effects or interaction effects on existing basis functions. The selected knots for the initial basis functions are also possible candidates for the new basis functions. The following Algorithm 4.8 explains the routine to create the MARS approximation. The following example demonstrates the method.

As in previous methods, the input parameters for the MARS approximation for iteration 0 are described in Table 4.1 and the generated function is displayed at the beginning of this chapter. For iteration 1, 10 more points are added. The updated input parameters are $N = 10 + 10 = 20$, $M_{max} = 10$ and $T = 3$. Once the algorithm reads the total points, it determines the set of eligible knots. Similar to sequential MARS 4, the initial knots (from iteration 0) are fixed for the rest of the iterations, allowing the ability to only use the initial selected candidate knots or to add more knots in the next iterations. The method for setting knots previously defined is also employed in this approach. Algorithm 4.5 is used to set the initial set of knots, Algorithms 4.6 and 4.7 are used if new knots are required to be set. Note that as in sequential MARS 4 method, this approach is limited to use in the MARS variants that require piecewise-linear fit only. After setting the knots, the data matrix x is standardized for numerical stability based on the midrange and half range from $N = 20$. The vector of response variables is read and $\bar{y}$ is computed. The knots from the existing MARS approximation (iteration 0) are based on the midrange and half range from $N = 10$, therefore, they need to be re-standardize based on the total number of points. As before, to obtain the actual value for the knots, calculate:

$$\text{knot} = (\text{knot} \cdot \text{half range}) + \text{midrange};$$

and re-standardize based on all the points:

$$\text{knot} = (\text{knot} - \text{new midrange})/\text{new half range}.$$

The β_m coefficients need to be updated following the same procedure described as in Section 4.2. The updated exiting MARS approximation for iteration 0 is:

$\hat{y}^{(1)} = \beta_0 + \beta_1 \cdot B_1 + \beta_2 \cdot B_2 + \beta_3 \cdot B_3 + \beta_4 \cdot B_4$

where,

$\beta_0 = 0.6698, \quad \beta_1 = -0.7543, \quad \beta_2 = -0.7228, \quad \beta_3 = -0.0257, \quad \beta_4 = -0.0522,$

$B_1 = [-(x_6 - 0.6849)]_+,$

$B_2 = [+(x_6 - 0.6849)]_+,$

$B_3 = [-\{\frac{- \cdot 0.6849}{(- \cdot 0.6849-1)} + \frac{+ \cdot 0.6849}{(+ \cdot 0.6849-1)} + \frac{- \cdot x_6}{(1- - \cdot 0.6849)} + \frac{+ \cdot x_{19}}{(1- + \cdot 0.6849)}\}]_+$, and

$B_4 = [+\{\frac{- \cdot 0.6849}{(- \cdot 0.6849-1)} + \frac{+ \cdot 0.6849}{(+ \cdot 0.6849-1)} + \frac{- \cdot x_6}{(1- - \cdot 0.6849)} + \frac{+ \cdot x_{19}}{(1- + \cdot 0.6849)}\}]_+.$

In order to generate the next MARS approximation (iteration 1), the Forward Coefficient Restriction (FCR) Algorithm which selects all potential basis functions needs to be initialized that is, all existing main and interaction terms need to be considered. Basically, it assigns the information of the existing MARS approximation to the forward algorithm format. The basis functions need to be orthonormalized that is, check if they are linearly independent and normalize them. After performing the tasks described, the new MARS approximations results as follow:

$\hat{y}^{(1)} = \beta_0{+}\beta_1{\cdot}B_1{+}\beta_2{\cdot}B_2{+}\beta_3{\cdot}B_3{+}\beta_4{\cdot}B_4\ \beta_5{\cdot}B_5{+}\beta_6{\cdot}B_6{+}\beta_7{\cdot}B_7{+}\beta_8{\cdot}B_8{+}\beta_9{\cdot}B_9{+}\beta_10{\cdot}B_{10}$

where,

$\beta_0 = -0.7765, \quad \beta_1 = 0.2850, \quad \beta_2 = -2.1301, \quad \beta_3 = 0.1003, \quad \beta_4 = -0.0251,$

$\beta_5 = 11.5752, \quad \beta_6 = -0.1113, \quad \beta_7 = 0.1223, \quad \beta_8 = 0.8831, \quad \beta_9 = -0.0019,$

$\beta_{10} = -0.2193,$

$B_1 = [-(x_6 - 0.6849)]_+,$

$B_2 = [+(x_6 - 0.6849)]_+,$

$B_3 = [-\{\frac{- \cdot 0.6849}{(- \cdot 0.6849-1)} + \frac{+ \cdot 0.6849}{(+ \cdot 0.6849-1)} + \frac{- \cdot x_6}{(1- - \cdot 0.6849)} + \frac{+ \cdot x_{19}}{(1- + \cdot 0.6849)}\}]_+,$

$B_4 = [+\{\frac{-\cdot 0.6849}{(-\cdot 0.6849-1)} + \frac{+\cdot 0.6849}{(+\cdot 0.6849-1)} + \frac{- \cdot x_6}{(1- - \cdot 0.6849)} + \frac{+ \cdot x_{19}}{(1- + \cdot 0.6849)}\}]_+,$

$B_5 = [-(\{\frac{- \cdot 0.6849}{(- \cdot 0.6849-1)} + \frac{- \cdot 0.4819}{(- \cdot 0.4819-1)} + \frac{- \cdot x_6}{(1- - \cdot 0.6849)} + \frac{- \cdot x_{17}}{(1- - \cdot 0.4819)}\}]_+,$

$B_6 = [+\{\frac{- \cdot 0.6849}{(- \cdot 0.6849-1)} + \frac{- \cdot 0.4819}{(- \cdot 0.4819-1)} + \frac{- \cdot x_6}{(1- - \cdot 0.6849)} + \frac{- \cdot x_{17}}{(1- - \cdot 0.4819)}\}]_+,$

$B_7 = [-\{\frac{- \cdot 0.6849}{(- \cdot 0.6849-1)} + \frac{+ \cdot 0.4819}{(+ \cdot 0.4819-1)} + \frac{- \cdot 0.4819}{(- \cdot 0.4819-1)} + \frac{- \cdot x_6}{(1- - \cdot 0.6849)} + \frac{+ \cdot x_{17}}{(1- + \cdot 0.4819)} +$
$\frac{- \cdot x_{18}}{(1- - \cdot -0.8252)}\}]_+,$

$B_8 = [+\{\frac{- \cdot 0.6849}{(- \cdot 0.6849-1)} + \frac{+ \cdot 0.4819}{(+ \cdot 0.4819-1)} + \frac{- \cdot 0.4819}{(- \cdot 0.4819-1)} + \frac{- \cdot x_6}{(1- - \cdot 0.6849)} + \frac{+ \cdot x_{17}}{(1- + \cdot 0.4819)} +$
$\frac{- \cdot x_{18}}{(1- - \cdot -0.8252)}\}]_+,$

$B_9 = [+\{\frac{- \cdot 0.6849}{(- \cdot 0.6849-1)} + \frac{- \cdot 0.6849}{(- \cdot 0.6849-1)} + \frac{- \cdot -0.9949}{(- \cdot -0.9949-1)} + \frac{- \cdot x_6}{(1- - \cdot 0.6849)} + \frac{- \cdot x_{19}}{(1- - \cdot 0.6849)} +$
$\frac{- \cdot x_3}{(1- - \cdot -0.9949)}\}]_+,$ and

$B_{10} = [-\{\frac{- \cdot 0.6849}{(- \cdot 0.6849-1)} + \frac{- \cdot 0.6849}{(- \cdot 0.6849-1)} + \frac{- \cdot -0.9949}{(- \cdot -0.9949-1)} + \frac{- \cdot x_6}{(1- - \cdot 0.6849)} + \frac{- \cdot x_{19}}{(1- - \cdot 0.6849)} +$
$\frac{- \cdot x_3}{(1- - \cdot -0.9949)}\}]_+.$

The final approximation has the two initial main effects B_1 and B_2, and the two initial two-way interaction effects B_3 and B_4. New interaction terms were formed, one pair of two-way interaction effects B_5 and B_6 having as a parent term B_1; and two pairs of three-way interaction effects. B_7 and B_8 having as a parent term B_6; and B_9 and B_{10} having as a parent term B_3. The new basis function has a total of 10 basis functions. Note that for the second iteration the maximum number of basis functions was $M_{max} = 5$ and it actually added 6 basis functions, this is because the number of selected basis functions for the first iteration was only 4 and the total number of basis functions for the new approximation was $M_{max} = 10$. Also, for the three-way interaction terms, the second split has the opposite sign from the two-way interaction parent term. For example the second split for basis function B_7 has a positive sign while its parent term B_6 has a negative sign. This is because the three way interaction term is really using ϕ which is positive. The term ϕ is only required for the last split of the basis function, for the last split term $s = \phi$ (Section 3.1.1). The refit of coefficients was based on a piecewise-linear fit. After finishing the MARS approximation, a testing data set is used to define the quality of fit based on the mean squared error (MSE).

Opposite to sequential MARS 3 and 4, this method can be used when the underlying function is known to have a significant number of interaction effects. This approach is the most comparable with sequential MARS 1, they both allow forming new interaction terms, however sequential MARS 5 is forced to maintain the basis functions from previous iterations as well as the eligible knots.

Algorithm 4.8 Sequential MARS 5 Algorithm

```
MARS approximation for iteration $i = 0$ has been previously created.
for all $i = 1, ..., iter$ iterations do
    define the set of new eligible knots ($T_i$),
    save the set of total eligible knots (total_knots $= T_i + T_{i-1}$) for next iteration [$i + 1$].
    follow Algorithm 4.2 to re-standardize selected knots $k^*$ and
    to update $\beta_{m^*}$ coefficients.
    for all $B_{m^*} = 1, ..., M^*$ do
        Add and orthonormalize basis function $B_M(x)$, $(m^*, x_{v^*}, k^*)$.
    end for $B_{m^*}$
    initialize $B_1(x) = B_{m^*} + 1$, $M = M^* + 1$, $LOF^* = LOF$.
    follow Algorithm 4.1 to generate new MARS approximation function based on new $N$, $M_{max}$ and total_knots,
    compute mean squared error (MSE) on testing data set.
end for $i$
```

CHAPTER 5

CASE STUDIES

5.1 Inventory Forecasting Problem

5.1.1 Convex MARS

Convex MARS was previously tested by Shih [1] on four-dimensional and nine-dimensional inventory forecasting stochastic dynamic programming (SDP) problems. Such application was initially studied by Chen [23] where a detailed explanation can be found. A brief description is given below.

The objective of the SDP is to minimize costs over T time periods which is represented by the following formulation:

$$\min_{u_1,\ldots,u_T} E\{\sum_{t=1}^{T} c_t(x_t, u_t, \epsilon_t)\} \tag{5.1}$$

$$\text{s.t.} \quad x_{t+1} = f(x_t, u_t, \epsilon_t) \quad \text{for} \quad t = 1, ..., T-1 \quad \text{and}$$

$$(x_t, u_t) \in \Gamma_t \quad \text{for} \quad t = 1, ..., T.$$

The cost function involves holding and backorder costs for each item, x_t represents the state vector which considers the inventory level of each item and its forecasted demand at the beginning of time period t, u_t represents the decision vector which is the amount of a particular item ordered in period t, ϵ_t is the stochastic component, x_{t+1} is determined by the transition function $f(\cdot)$ which involves the change in the forecast for a particular period from the mean demand for that period and Γ_t represents capacity constraints.

The future value function provides the optimal cost to operate the system from period t through T as a function of the state vector x_t. This is written in summation form as:

$$F_t(x_t) = \min_{u_1,\ldots,u_T} E \sum_{\tau=t}^{T} c_\tau(x_\tau, u_\tau, \epsilon_\tau)$$

$$\text{s.t.} \quad x_{\tau+1} = f(x_\tau, u_\tau, \epsilon_\tau) \quad \text{for} \quad \tau = t, ..., T-1 \quad \text{and}$$

$$(x_\tau, u_\tau) \in \Gamma_\tau \quad \text{for} \quad \tau = 1, ..., T.$$

In theory the optimal value function is known to be convex, therefore convex MARS is applied to fit the data in the last time period of the three-period inventory forecasting SDP.

5.1.1.1 Four-dimensional Inventory Forecasting Problem

The four covariates for this problem represent two different products and their corresponding demand forecast. An orthogonal array (OA) of strength three ($d = 3$) and levels $p = 5$ was created, resulting $N = p^d = 125$ discretization points. The response variable represents the actual cost value from the last stage in the dynamic program. Original MARS, convex MARS with $\tau = 0$ and convex MARS with $\tau > 0$ models were generated considering a maximum number of basis functions of $M_{max} = 100$ and were restricted up to three-way interactions. Three of the models used the automatic stopping rule (ASR) (Tsai and Chen [21]). Original MARS only used the forward algorithm. The threshold was defined based on the strategy proposed by Shih [1] described in Section 2.1.2. The threshold used represents 6.43% of the largest absolute coefficient of the original MARS function which is 357.61. Table 5.1 shows the parameters considered for each of the models.

Table 5.1. Parameter settings for the versions of MARS on the four-dimensional inventory forecasting problem.

	Original MARS	Convex MARS ($\tau = 0$)	Convex MARS ($\tau > 0$)
M_{max}		100	
Knots		3	
Interactions		3	
ASR		2	
ASR difference		0.02	
Robust	1	0	0
Robust tolerance	0.3	N/A	N/A
Threshold	N/A	0.00	23.00

A validation data set of 100 points with a mean of 137.47 was used to test the performance of the models. Figure 5.1 shows the boxplots of the absolute error defined by $|y - \hat{f}|$, where y is the actual cost and $\hat{f}$ is the estimated cost obtained from the approximation model.

5.1.1.2 Nine-dimensional Inventory Forecasting Problem

The nine covariates for this problem represent three different products and two demand forecasts for each of them. An orthogonal array (OA) of strength three ($d = 3$) and levels $p = 11$ was created, resulting $N = p^d = 1,331$ discretization points. The response variable represents the actual cost value from the last stage in the dynamic program. Original MARS, convex MARS with $\tau = 0$ and convex MARS with $\tau > 0$ models were generated considering a maximum number of basis function of $M_{max} = 300$ and were restricted up to three-way interactions. Three of the models used the automatic stopping rule (ASR) (Tsai and Chen [21]). Original MARS only used the forward algorithm. The threshold used represents 2.35% of the largest absolute coefficient of the original MARS function which is 1,399.32. Table 5.2 shows the parameters considered for each of the models.

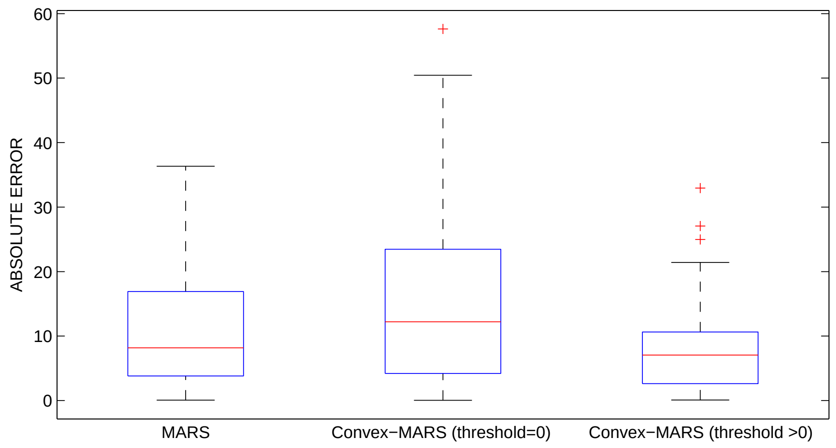


Figure 5.1. Four-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 100 points: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$).

Table 5.2. Parameter settings for the versions of MARS on the nine-dimensional inventory forecasting problem.

	Original MARS	Convex MARS ($\tau = 0$)	Convex MARS ($\tau > 0$)
M_{max}		300	
Knots		9	
Interactions		3	
ASR		2	
ASR difference		0.02	
Robust	1	0	0
Robust tolerance	0.3	N/A	N/A
Threshold	N/A	0.00	31.50

A validation data set of 1,000 points with a mean of 376.33 was used to test the performance of the models. Figure 5.2 shows the boxplots of the absolute error which is defined in the same way as previously described.

For the four and nine dimensional cases, the performance of convex MARS is comparable to the original MARS model. From Figure 5.1, it can be observed that the

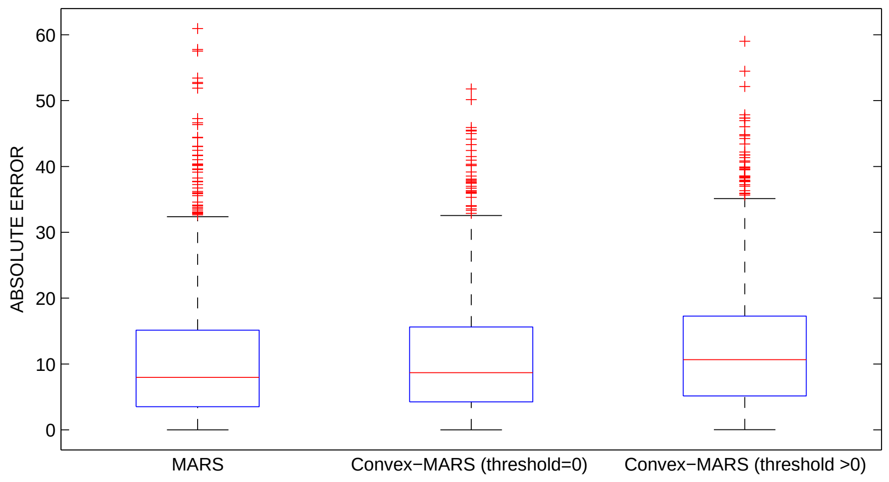


Figure 5.2. Nine-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 1000 points: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$).

median of the absolute error of the convex MARS model is higher than the original MARS, and the median of the absolute error using a threshold (convex MARS with $\tau > 0$) is slightly lower than the original MARS indicating a better fit. However, the convex versions guarantee convexity which is the purpose of the method. From Figure 5.2, the median of the absolute error for the convex versions is slightly higher than the median of the original MARS, but it is important to consider that they are comparable and more importantly, the models assure convexity.

Another benefit of the convex MARS models is that the complexity of the models is reduced compared to the original MARS, i.e. the number of selected basis functions is smaller. For example, the original MARS model for the four-dimensional case selected 39 basis functions while the convex MARS with $\tau = 0$ model only selected 10 basis functions. The convex MARS with $\tau > 0$ model selected 18, which is also significantly smaller. For the nine-dimensional case, 45 basis functions were selected for the original MARS model while the convex MARS with $\tau = 0$ model selected 17 basis functions and the convex MARS with $\tau > 0$ model selected 15 basis functions.

In theory, for a true underlying additive convex function, original MARS can create a convex approximation, but it may use a mix of convex and nonconvex univariate pairs, adding unnecessary complexity to the model. Tables 5.3 and 5.4 contain the details of the main effect terms of the original MARS function for the four-dimensional and nine-dimensional inventory forecasting problems respectively. From the tables, it can be observed that there are positive and negative β_m coefficients for unpaired basis functions and positive and negative sum of β_m coefficients for pair basis functions. For example in Table 5.3, the unpaired basis function $m = 11$, has a positive coefficient, while the unpaired basis function $m = 18$ has a negative coefficient. Basis function $m = 1$ and $m = 2$ form a pair and the sum of their coefficients is positive. In the same manner, basis function $m = 27$ and $m = 28$ form a pair but the sum of the coefficients is negative. Similarly, for the nine-dimensional case in Table 5.4, the unpaired basis function $m = 41$ has a positive coefficient however, in this case there are not unpaired basis functions with a negative coefficient. Basis function $m = 3$ and $m = 4$ form a pair and the sum of their coefficients is positive. In the same manner, basis function $m = 1$ and $m = 2$ form a pair but the sum of the coefficients is negative. Convex MARS guarantees convexity selecting only the convex terms. Consequently the number of terms for original MARS will likely be higher than convex MARS because, convex MARS is more efficient in selecting the final basis functions.

Table 5.5 shows different values for the threshold that were tested for both the four-dimensional and the nine-dimensional inventory forecasting problems. They were defined based on various percentages of the maximum absolute coefficient of the original MARS approximation function. For the four-dimensional case, the minimum

Table 5.3. Main effect terms of the original MARS function for the four-dimensional inventory forecasting problem.

m	β_m	$s_{l,m}$	$x_{v(l,m)}$	$k^-_{l,m}$	$k_{l,m}$	$k^+_{l,m}$
1	198.0000	-1	2	-0.25	0.00	0.38
2	14.6411	1	2	-0.25	0.00	0.50
3	120.9767	-1	1	-0.25	0.00	0.25
4	20.5489	1	1	-0.25	0.00	0.25
11	291.1943	-1	2	-0.75	-0.50	-0.25
18	-20.9203	-1	1	0.25	0.50	0.75
27	0.7903	-1	4	-0.50	0.00	0.50
28	-11.5013	1	4	-0.50	0.00	0.50
33	102.8038	-1	1	-0.75	-0.50	-0.25

Table 5.4. Main effect terms of the original MARS function for the nine-dimensional inventory forecasting problem.

m	β_m	$s_{l,m}$	$x_{v(l,m)}$	$k^-_{l,m}$	$k_{l,m}$	$k^+_{l,m}$
1	154.2599	-1	2	-0.30	0.40	0.50
2	-216.7826	1	2	0.25	0.40	0.50
3	391.2438	-1	1	-0.50	0.00	0.50
4	-210.2493	1	1	-0.50	0.00	0.50
5	213.9986	-1	3	-0.50	0.00	0.50
6	47.3307	1	3	-0.50	0.00	0.50
7	-39.3492	-1	5	-0.50	0.00	0.50
8	104.4757	1	5	-0.50	0.00	0.50
9	-26.7381	-1	4	-0.60	-0.20	0.40
10	96.2213	1	4	-0.60	-0.20	0.40
23	-34.8271	-1	6	-0.50	0.00	0.50
24	17.4784	1	6	-0.50	0.00	0.50
41	264.6705	-1	2	0.50	0.60	0.75

value for the median absolute error was obtained in a range of 8.39% to 10.00%. For the nine-dimensional case, the minimum value for the median absolute error was obtained in a range of 1.12% to 1.49%.

To extend the performance tests of the proposed method, random noise was added to the data sets of the two different cases, four and nine dimensions. The

Table 5.5. Comparison of various threshold values based on different percentages of the maximum absolute coefficient from original MARS. Median absolute error is reported.

Four-dimensional			Nine- dimensional		
Percentage	Threshold	Median	Percentage	Threshold	Median
2.00	7.15	7.19	1.12	15.00	8.68
5.00	17.88	7.19	1.49	20.00	8.68
5.59	20.00	6.89	1.57	21.00	8.94
5.87	21.00	7.07	1.61	21.5	8.94
6.00	21.45	7.07	1.72	23.00	9.07
6.43	23.00	7.07	1.87	25.00	20.46
7.00	25.03	7.07	2.00	26.78	20.46
8.00	28.60	7.07	2.24	30.00	11.10
8.39	30.00	6.23	2.61	35.00	39.79
9.00	32.18	6.23	5.00	66.96	37.20
9.79	35.00	6.23	8.00	107.14	37.63
10.00	35.76	6.23	10.00	133.93	35.58

random noise was generated following a normal distribution considering a coefficient of variation (CV) of 0.01, 0.05 and 0.10, adjusting the standard deviation with their corresponding mean (CV $= \frac{\sigma}{\mu}$) for both, training and testing data sets. Figures 5.3 to 5.8 show the boxplots of the absolute error for all the cases.

The percentages shown in Table 5.5 were also tested in order to define the threshold for convex MARS that yields the best result, in this case the minimum median absolute error (Tables 5.6 and 5.7). For the 4-dimensional case using the data set with noise based on CV=0.01, the threshold that gives the best result is between 6.43% and 9.00% of the largest absolute coefficient from the original MARS function (297.32). The median absolute error varies from a range of 90.46 to 91.04. This percentage range is comparable to the percentage range (8.39% to 10.00%) that gives the best result using the original data (Table 5.5). For the data set with noise based on CV=0.05, the percentage that gives the best result is 2.00% from the largest absolute coefficient from the original MARS function (412.02) and the median absolute error varies from 90.61 to 90.83. For the data set with noise based on CV=0.10, the percentage range that provides the best result is from 6.00% to 6.43% from the largest absolute coefficient from the original MARS function (647.04), and the median varies from 92.94 to 96.47.

Table 5.6. Four-dimensional case. Comparison of various threshold values based on different percentages of the maximum absolute coefficient from original MARS tested in data sets containing random noise. Median absolute error is reported.

	CV=0.01		CV=0.05		CV=0.10	
Percentage	Threshold	Median	Threshold	Median	Threshold	Median
2.00	5.95	91.04	8.24	90.61	12.94	93.13
5.00	14.87	91.04	20.60	90.62	32.35	93.13
5.59	16.62	91.04	23.03	90.62	36.17	96.47
5.87	17.45	91.04	24.19	90.62	37.98	96.47
6.00	17.84	91.04	24.72	90.62	38.82	92.94
6.43	19.12	90.46	26.49	90.62	41.60	92.94
7.00	20.81	90.46	28.84	90.62	45.29	96.13
8.00	23.79	90.46	32.96	90.62	51.76	96.06
8.39	24.95	90.46	34.57	90.62	54.29	96.06
9.00	26.76	90.46	37.08	90.62	58.23	96.06
9.79	29.11	90.73	40.34	90.83	63.35	95.79
10.00	29.73	90.73	41.20	90.83	64.70	95.79

For the 9-dimensional case (Table 5.7), using the data set with noise based on CV=0.01, the threshold that gives the best result is 1.87% from the largest absolute coefficient of the original MARS function (1,306.82) and the median absolute error varies from 32.30 to 56.11. This percentage is comparable to the percentage range (1.12% to 1.49%) that gives the best threshold using the original data (Table 5.5). For the data set with noise based on CV=0.05, the percentage from the largest absolute coefficient from the original MARS function (440.98) that gives the best result is from 7.00% to 7.14%, and the median absolute error varies from 32.77 to 37.66 (this percentage is not shown in Table 5.7). For the data set with noise based on CV=0.10 the percentage from the largest absolute coefficient from the original MARS function (495.86) that gives the best result is 8.00%, varying from 43.52 to 47.70. The boxplots for the convex MARS with $\tau > 0$ from Figures 5.3 to 5.8 used the threshold that yields the best result.

Table 5.7. Nine-dimensional case. Comparison of various threshold values based on different percentages of the maximum absolute coefficient from original MARS tested in data sets containing random noise. Median absolute error is reported.

	CV=0.01		CV=0.05		CV=0.10	
Percentage	Threshold	Median	Threshold	Median	Threshold	Median
1.12	14.64	36.50	4.94	37.40	5.55	43.55
1.49	19.47	36.50	6.57	37.40	7.39	43.55
1.57	20.52	36.50	6.92	37.40	7.79	43.55
1.61	21.04	36.50	7.10	37.40	7.98	43.55
1.72	22.48	36.50	7.58	37.40	8.53	43.55
1.87	24.44	32.30	8.25	37.40	9.27	43.55
2.00	26.14	32.91	8.82	37.40	9.92	43.55
2.24	29.27	36.50	9.88	37.40	11.11	43.55
2.61	34.11	46.95	11.51	37.40	12.94	43.55
5.00	65.34	41.09	22.05	37.40	24.79	43.55
8.00	104.55	46.52	35.28	37.66	39.67	43.52
10.00	130.68	56.11	44.10	37.66	49.59	47.70

5.1.1.3 Conclusions

Figure 5.9 and Figure 5.10 show the median absolute error for original MARS, convex MARS with $\tau = 0$ and convex MARS with $\tau > 0$ for the different data sets for the four-dimensional and nine-dimensional cases respectively. Table 5.8 and Table 5.9 contain the numerical results.

Table 5.8. Median of the absolute error (four-dimensional case).

	Original data	CV=0.01	CV=0.05	CV=0.10
Original MARS	8.19	94.63	91.44	95.00
Convex MARS with $\tau = 0$	12.22	90.83	91.82	153.59
Convex MARS with $\tau > 0$	7.07	90.46	90.61	92.94

Table 5.9. Median of the absolute error (nine-dimensional case).

	Original data	CV=0.01	CV=0.05	CV=0.10
Original MARS	7.97	31.94	34.24	40.02
Convex MARS with $\tau = 0$	8.68	36.50	37.40	43.55
Convex MARS with $\tau > 0$	10.65	32.30	32.77	43.52

Overall, the results of convex MARS with $\tau = 0$ and convex MARS with $\tau > 0$ are comparable to the original MARS method. However, by restricting the selection of basis functions with a threshold greater than zero (convex MARS with $\tau > 0$) the accuracy of the approximation model is higher. For the four-dimensional case, all the medians from the convex MARS with $\tau > 0$ models are lower than the medians from the original MARS models. This is not the case for the medians from the convex MARS without threshold ($\tau = 0$), since only the median with data considering CV=0.01 for the random noise is lower than the original MARS. For the nine-dimensional case, almost all the medians from original MARS models are lower than the medians from convex MARS with $\tau = 0$ and convex MARS with $\tau > 0$ except for the median of convex MARS with $\tau > 0$ using the data considering CV=0.05 for the random noise. However, the convex MARS with $\tau = 0$ and convex MARS with $\tau > 0$ models guarantee convexity. And for this inventory forecasting SDP application convexity must be assured to obtain the global optimum. The strategy to select the best threshold is still under research, the convex MARS approximation function could be very sensitive to the threshold value, significantly affecting the results. Having a threshold too high might force the convex MARS algorithms [1] to select very few basis functions degrading the quality of fit of the function. In contrast, if the threshold is too low, the results might not be the best. In terms of CPU time, all the tested MARS runs required less than 5 seconds on a Quad 3.00-GHz 8GB RAM Dell Precision Workstation.

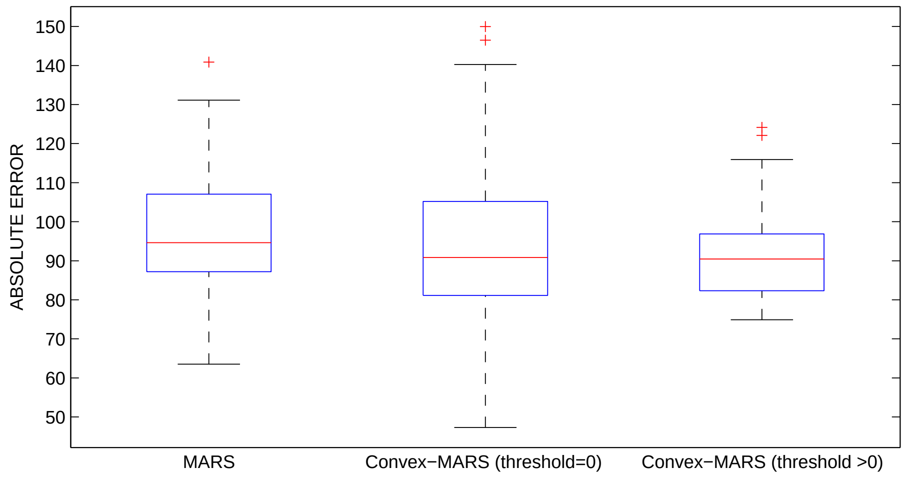


Figure 5.3. Four-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 100 points. Data sets include random noise, CV=0.01: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$).

5.1.2 MARS Variants

The SDP inventory forecasting problem for four and nine dimensions was also applied to the different variants of MARS explained in Section 3.2. The different variants are:

1. Original MARS
2. Nonconvex piecewise-linear

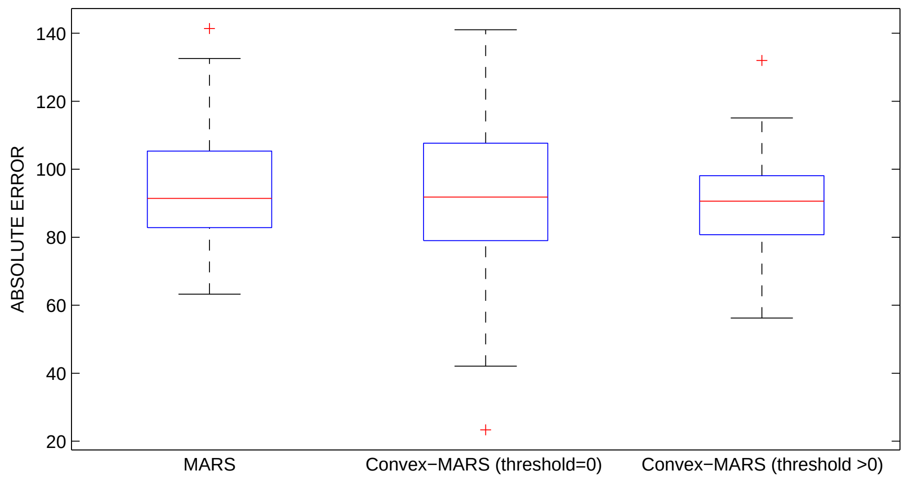


Figure 5.4. Four-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 100 points. Data sets include random noise, CV=0.05: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$).

3. Nonconvex piecewise-linear using original backward algorithm
4. Nonconvex smooth
5. Nonconvex smooth using original backward algorithm
6. Convex piecewise-linear
7. Convex smooth

The parameters used to generate the different models are described in Tables 5.10 and 5.11 for the four and nine dimensional cases respectively. The threshold used was the percentage that provided the best results based on Table 5.5 ($\tau = 30$ for the four dimensional case and $\tau = 15$ for the nine dimensional case).

Figures 5.11 and 5.12 illustrate the boxplots for the different MARS models for the four and nine dimensional cases respectively. Tables 5.12 and 5.13 show the numerical results for the boxplots.

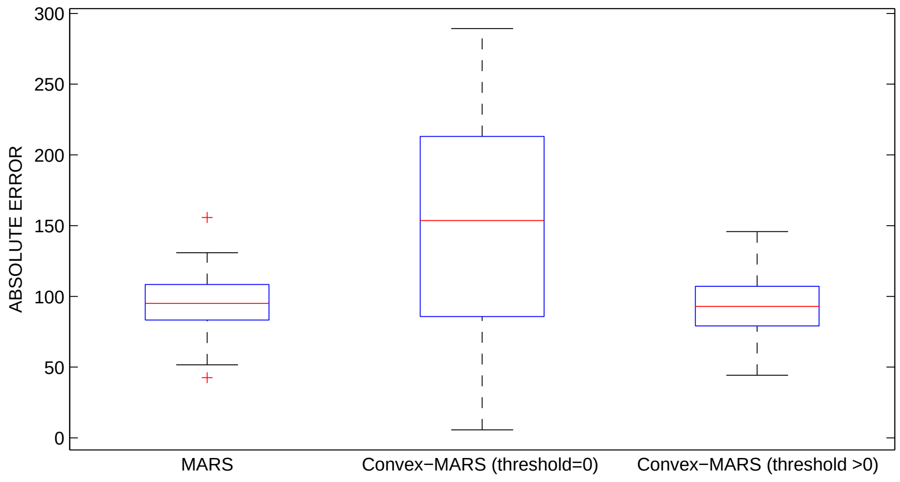


Figure 5.5. Four-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 100 points. Data sets include random noise, CV=0.10: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$).

Table 5.10. Parameter settings for MARS variants on the four-dimensional inventory forecasting problem.

	1	2	3	4	5	6	7
M_{max}				100			
Knots				3			
Interactions				3			
ASR				2			
ASR difference				0.02			
Robust	1	0	0	0	0	0	0
Robust tolerance	0.3	N/A	N/A	N/A	N/A	N/A	N/A
Original backward algorithm	0	0	1	0	1	N/A	N/A
Convex	N/A	0	0	0	0	1	1
Threshold	N/A	N/A	N/A	N/A	N/A	30.00	30.00
Smooth	0	0	0	1	1	0	1

5.1.2.1 Conclusions

For both cases, four and nine dimensions, the performance of MARS variants is comparable to original MARS. However, a comparison between the nonconvex and

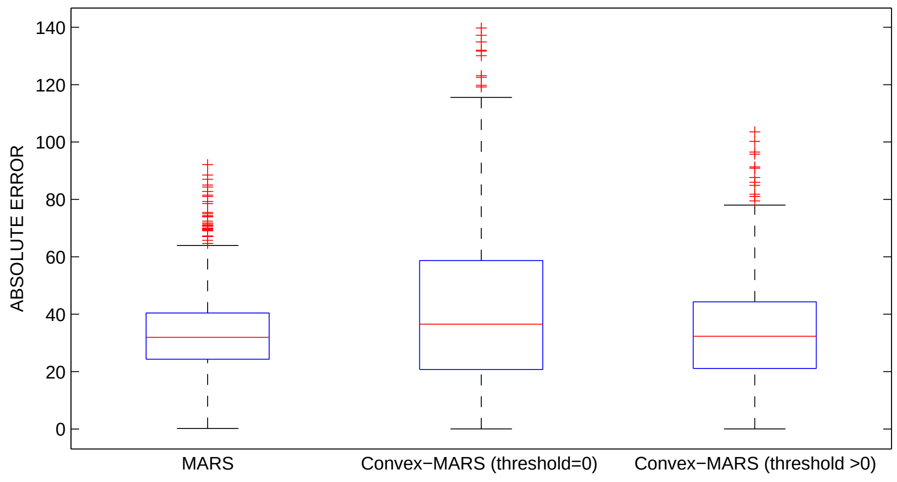


Figure 5.6. Nine-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 1000 points. Data sets include random noise, CV=0.01: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$).

Table 5.11. Parameter settings for MARS variants on the nine-dimensional inventory forecasting problem.

	1	2	3	4	5	6	7
M_{max}				300			
Knots				9			
Interactions				3			
ASR				2			
ASR difference				0.02			
Robust	1	0	0	0	0	0	0
Robust tolerance	0.3	N/A	N/A	N/A	N/A	N/A	N/A
Original backward algorithm	0	0	1	0	1	N/A	N/A
Convex	N/A	0	0	0	0	1	1
Threshold	N/A	N/A	N/A	N/A	N/A	15.00	15.00
Smooth	0	0	0	1	1	0	1

convex models might not be fair since the accuracy of the prediction model depends on the input data and the properties of the underlying function. In this case, the data and the underlying function are known to be convex, therefore there is probably

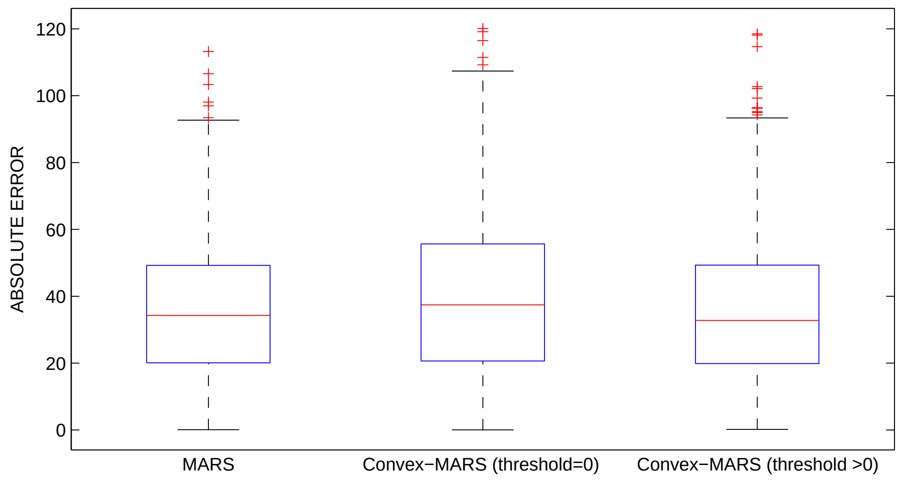


Figure 5.7. Nine-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 1000 points. Data sets include random noise, CV=0.05: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$).

no benefit to force a convex approximation. The advantage of using the original backward algorithm in the nonconvex models is that, it can reduce the complexity of the model and still have a good fit. The purpose of MARS variants is to provide more flexibility to estimate the function according to the properties of the underlying true function.

Table 5.12. MARS variants absolute error boxplot numerical results on the four-dimensional inventory forecasting problem.

MARS variants	1	2	3	4	5	6	7
Maximum	36.34	34.91	48.05	25.82	41.12	37.71	33.00
3rd Quartile	16.85	13.86	15.18	10.98	13.54	11.08	11.04
Median	8.19	4.94	8.65	6.02	6.32	5.88	6.23
1st Quartile	3.89	1.80	3.89	3.63	3.17	2.66	2.56
Minimum	0.07	0.24	0.04	0.13	0.24	0.12	0.03

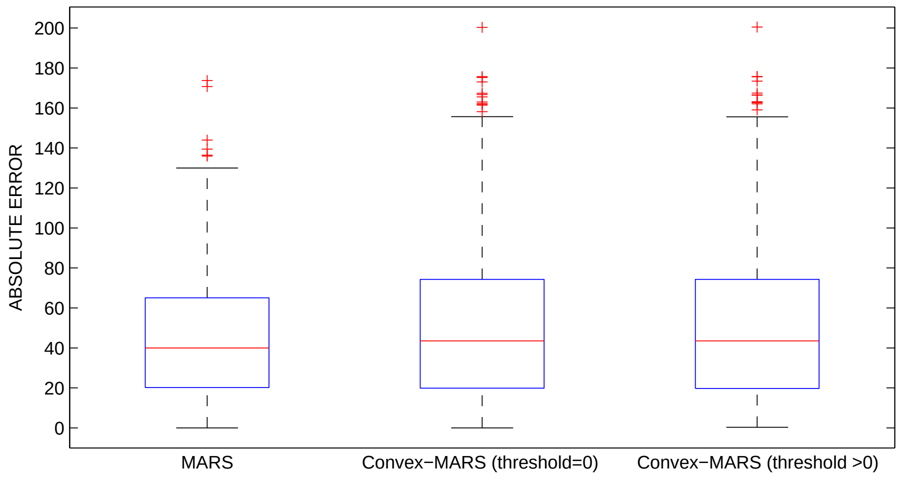


Figure 5.8. Nine-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 1000 points. Data sets include random noise, CV=0.10: (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$).

Table 5.13. MARS variants absolute error boxplot numerical results on the nine-dimensional inventory forecasting problem.

MARS variants	1	2	3	4	5	6	7
Maximum	60.94	58.61	58.61	54.84	54.84	56.03	51.78
3^{rd} Quartile	15.14	15.40	15.40	14.78	14.78	16.03	15.61
Median	7.97	7.80	7.80	8.15	8.15	8.93	8.68
1^{st} Quartile	3.52	3.57	3.57	3.74	3.74	4.60	4.26
Minimum	0.00	0.01	0.01	0.00	0.00	0.04	0.01

5.2 Safety System Design of Automotive Vehicle: Nonconvex Piecewise-Linear MARS with Binary Variables

The objective of this case of study is to optimize a flexible statistical function to minimize the impact of an automotive crash. This section is only focused on the development of the statistical model. The full data design contains:

- 33 input variables (10 categorical and 23 continuous)

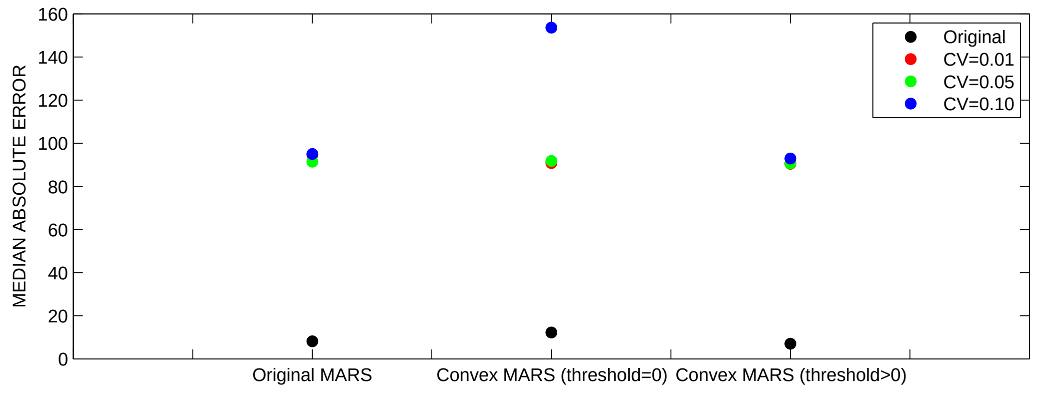


Figure 5.9. Four-dimensional inventory forecasting problem. Comparison of median absolute error from different data sets (original and with noise, CV=0.01, CV=0.05 and CV=0.10). (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$).

- 51 output variables (the objective and 50 constraints)
- 200 data points for training and 1,249 data points for testing.

Table 5.14 contains the description of each variable and the minimum and maximum values for the training and testing data sets, along with the number of levels.

From Table 5.14, it can be observed that there are seven categorical variables with levels $p = 2$, these are flag variables x_3, x_4, x_5, x_6, x_7, x_8, x_9. There are three categorical variables with $p > 2$, x_1 ($p = 4$), x_{22} ($p = 3$) and x_{33} ($p = 3$); for these cases the variable will be represented by $p - 1$ binary variables (dummy variables). The levels for continuous variables varies from $p = 2$ to $p = 7$. A discrepancy exists between the two data sets training and testing, regarding the number of levels and the range of variables. There are several variables that are missing at least one level in the training data, this could possibly lead to a poor fit for the testing set. The total number of input variables is 37, including two dummy variables for x_1, one dummy variable for x_{22} and one for x_{33}.

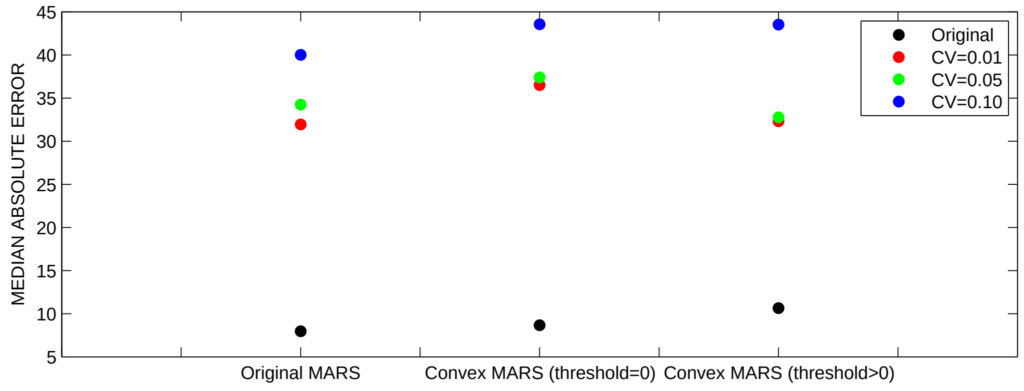


Figure 5.10. Nine-dimensional inventory forecasting problem. Comparison of median absolute error from different data sets (original and with noise, CV=0.01, CV=0.05 and CV=0.10). (1) MARS, (2) Convex MARS ($\tau = 0$), (3) Convex MARS ($\tau > 0$).

SAS software was used to generate multiple linear regression models to analyze the behavior of the data. Table 5.15 shows the R-squared for each model; y_1 represents the model for the objective variable while the rest of the responses represent the models for the constraints. The column curvature shows an indication to use a nonlinear model based on the residual plots if it exists.

From Table 5.15, 10 models appear to have some curvature, meaning that a linear model might not be the best option to represent the function, suggesting the usage of a flexible model instead. Original MARS might be an option since one of its main advantages is the ability to handle curvature. However, to facilitate the optimization process, the piecewise-linear version of MARS is recommended. The R-squared should improve for these cases. For the models that do not show curvature, there are many that have a low R-squared (R-sq $<$ 0.70). Consequently, a deeper study of the data design needs to be conducted. Figure 5.13 displays the residual plots for the cases that show curvature.

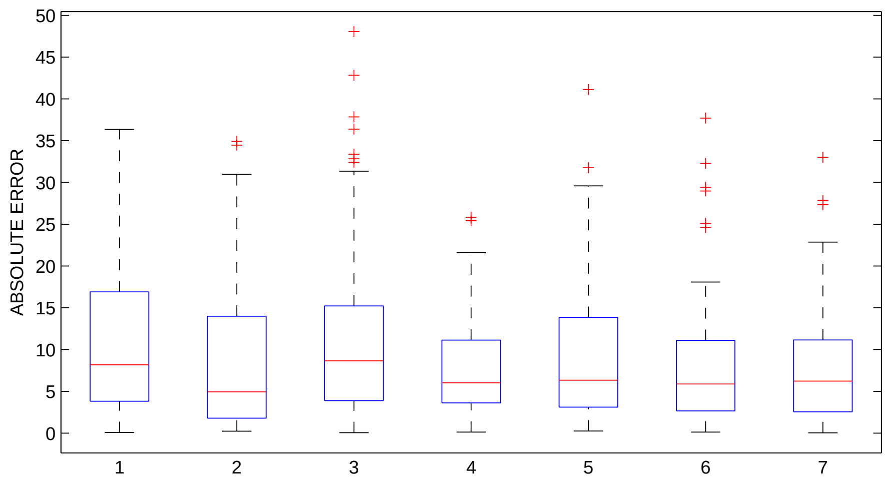


Figure 5.11. Four-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 100 points. (1) Original MARS, (2) Nonconvex piecewise-linear, (3) Nonconvex piecewise-linear using original backward algorithm, (4) Nonconvex smooth, (5) Nonconvex smooth using original backward algorithm, (6) Convex piecewise-linear ($\tau > 0$) and (7) Convex smooth ($\tau > 0$).

The input data were generated from a deterministic genetic algorithm. To see if correlation exists between the input variables, the variance inflation factor (VIF) was calculated. This index measures how much the variance of an estimated regression coefficient is increased because of collinearity. A common criterion to determine if multicollinearity is high is if VIF > 5. However, in an orthogonal (uncorrelated) experimental design, the VIF is equal to 1. From Table 5.16 it can be stated that the regressors have low variance inflation. The highest VIF value is 1.81054 for variable x_{22c} (dummy variable), but it is still considered low. Therefore, there is no strong multicollinearity between regressors.

An indicator to see if the data come from a good design is to review the minimum distance between the points in the design space and compare the result with the minimum distance between points coming from robust designs. Sobol and Latin

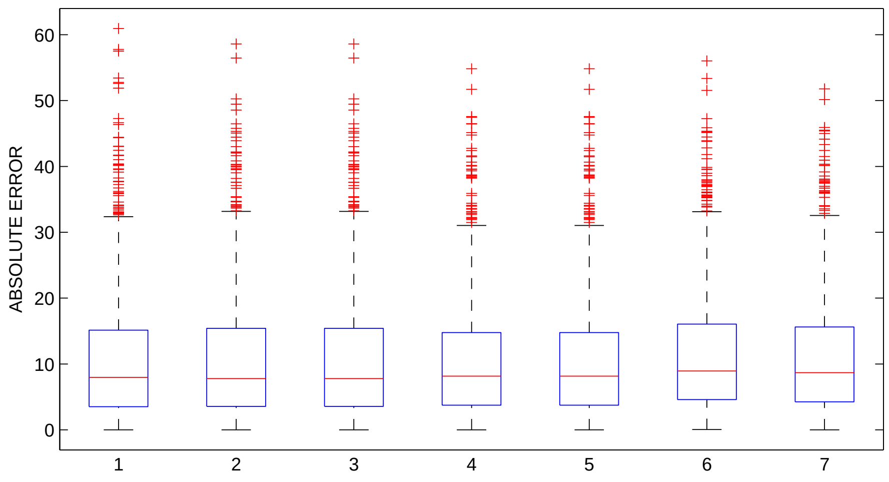


Figure 5.12. Nine-dimensional inventory forecasting problem. Comparison of boxplots based on a validation set of 100 points. (1) Original MARS, (2) Nonconvex piecewise-linear, (3) Nonconvex piecewise-linear using original backward algorithm, (4) Nonconvex smooth, (5) Nonconvex smooth using original backward algorithm, (6) Convex piecewise-linear ($\tau > 0$) and (7) Convex smooth ($\tau > 0$).

Hypercube (LHD) designs are considered low-discrepancy sequences. Thus, three different designs were created considering 37 variables and 200 points and a design range of [0.0001,0.9999]. Table 5.17 shows the minimum distance between points from the original data, Sobol design, LHD and Uniform design.

From Table 5.17, it is clear that the minimum distance between points in the original data is significantly low in comparison with any of the other designs. This means that the 200 input data points are not well spread out in the space and they might not be covering the whole region under study. This explains the lack of accuracy of a linear model for many of the models (Table 5.15). As an alternative, different data sets were created using the existing data (200 points for testing and 1,249 points for training) with the purpose to see if they could better represent the region under

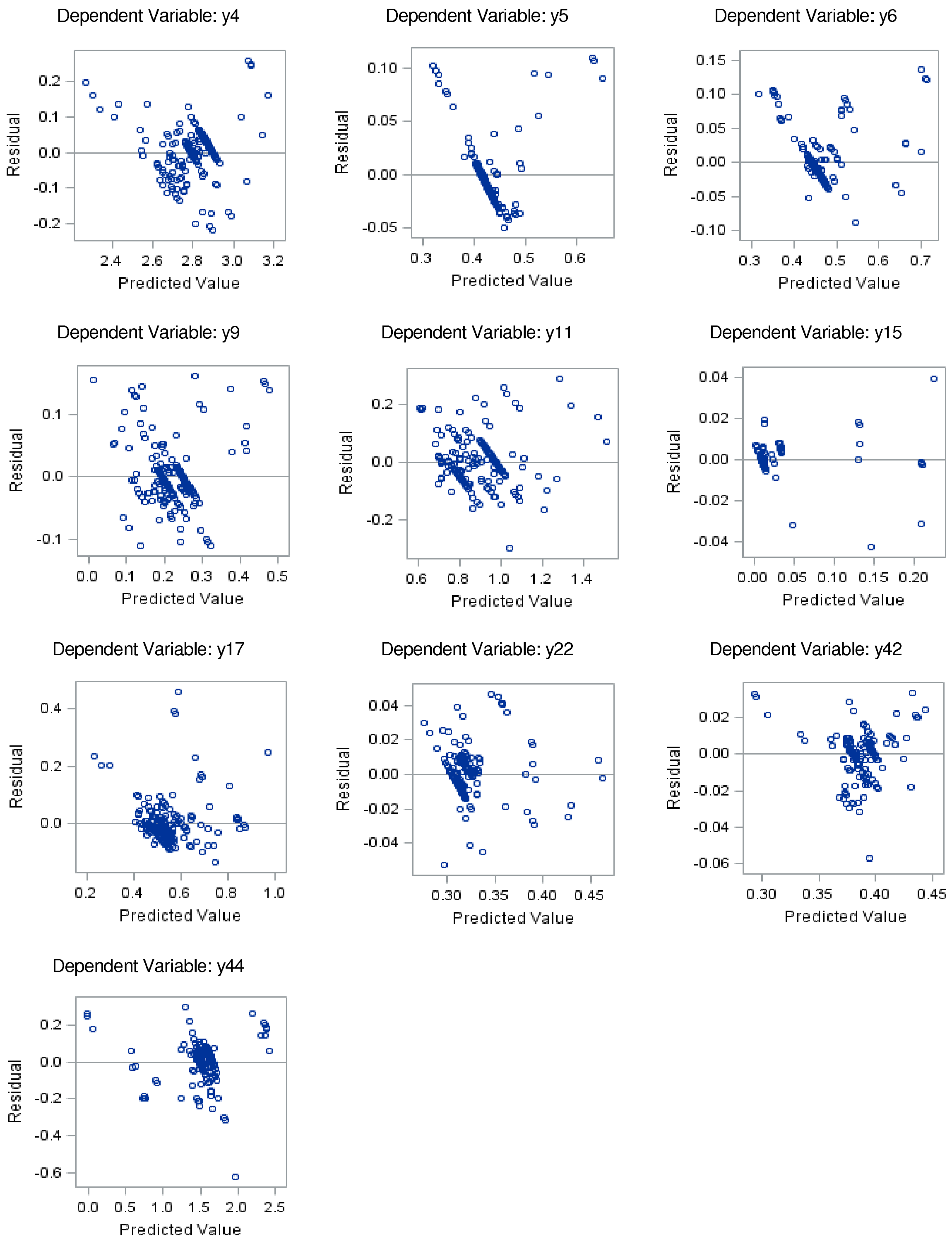


Figure 5.13. Residual plots of the models that showed curvature of the automotive vehicle case.

Table 5.14. Description and details of the input variables of the automotive vehicle case.

		Training data set			Testing data set		
ID	Variable description	Levels	Min	Max	Levels	Min	Max
x_1	PAB Shape	4	1	4	4	1	4
x_2	PAB Size	7	-0.1	0.9	13	-0.2	1
x_3	Buckle pretensioner flag	2	0	1	2	0	1
x_4	Retractor pretensioner flag	2	0	1	2	0	1
x_5	Adaptive belt load limiter flag	2	0	1	2	0	1
x_6	Crash locking tongue flag	2	0	1	2	0	1
x_7	Knee airbag flag	2	0	1	2	0	1
x_8	Passenger airbag adapt vent flag	2	0	1	2	0	1
x_9	Heel stopper flag	2	0	1	2	0	1
x_{10}	Buckle pretensioner pull in (m)	3	0.06	0.1	3	0.06	0.1
x_{11}	Buckle pretensioner time to fire (s)	2	0.008	0.013	2	0.008	0.013
x_{12}	Retractor pretensioner pull in (m)	2	0.06	0.08	3	0.06	0.1
x_{13}	Retractor pretensioner time to fire (s)	2	0.008	0.013	2	0.008	0.013
x_{14}	Retractor torsion bar force level	4	2000	2800	6	2000	3000
x_{15}	Retractor torsion bar force level	4	2000	3200	4	2000	3200
x_{16}	Retractor torsion bar displacement interval	3	0.05	0.2	6	0.05	0.3
x_{17}	Retractor torsion bar displacement interval	3	0.05	0.3	6	0.05	0.3
x_{18}	Knee airbag time to fire (s)	2	0.013	0.2	2	0.013	0.2
x_{19}	Knee airbag inflator power	3	0.75	1.5	4	0.75	1.5
x_{20}	Knee airbag vent size (mm)	2	0	15	2	0	15
x_{21}	Passenger airbag lower tether length (mm)	2	0.4	0.52	3	0.4	0.52
x_{22}	Passenger airbag lower tether location	3	1	4	4	1	4
x_{23}	Passenger airbag time to fire (s)	2	0.01	0.013	2	0.01	0.013
x_{24}	Passenger airbag Z-Scale	6	0.8	1.15	9	0.8	1.2
x_{25}	Passenger airbag adaptive vent size (mm)	4	40	120	5	40	120
x_{26}	Passenger airbag time to fire (s)	5	0.02	0.08	10	0.01	0.1
x_{27}	Passenger airbag time to fire (s)	5	0.02	0.08	10	0.01	0.1
x_{28}	Passenger airbag time to fire (s)	4	0.04	0.1	5	0.02	0.1
x_{29}	Passenger airbag time to fire (s)	3	0.02	0.08	5	0.02	0.1
x_{30}	Passenger airbag fixed vent size (mm)	3	40	70	5	40	80
x_{31}	Passenger airbag inflator power	3	0.8	1.2	5	0.8	1.2
x_{32}	Passenger airbag upper tether length (mm)	2	0.4	0.52	3	0.4	0.52
x_{33}	Passenger airbag upper tether location	3	2	4	4	1	4

study. First, based on the information from Table 5.14, 78 points were selected from the testing data set assuring that all the missing levels were present. These 78 points were added to the original 200 points, having a total of 278 points for training. The

Table 5.15. R-squared of the multiple linear regression models of the automotive vehicle case.

ID	Response variable name	Curvature	R-sq	ID	Response variable name	Curvature	R-sq
y_1	Obj_pb05_RRS		0.7771	y_{27}	constr_pb50_HIC		0.7222
y_2	constr_far50_ChestD		0.5735	y_{28}	constr_pb50_Head_IP_min		0.8508
y_3	constr_far50_ChestG		0.3797	y_{29}	constr_pb50_NeckFzMax		0.5842
y_4	constr_far50_Chest_IP_min	Yes	0.7572	y_{30}	constr_pb50_NeckFzMin		0.6575
y_5	constr_far50_FemurL	Yes	0.6520	y_{31}	constr_pb50_Nij		0.6330
y_6	constr_far50_FemurR	Yes	0.7043	y_{32}	constr_pu05_ChestD		0.4454
y_7	constr_far50_HIC		0.4811	y_{33}	constr_pu05_ChestG		0.8401
y_8	constr_far50_Head_IP_min		0.7845	y_{34}	constr_pu05_Chest_IP_min		0.8803
y_9	constr_far50_NeckFzMax	Yes	0.6279	y_{35}	constr_pu05_FemurL		0.5596
y_{10}	constr_far50_NeckFzMin		0.5591	y_{36}	constr_pu05_FemurR		0.5691
y_{11}	constr_far50_Nij	Yes	0.7191	y_{37}	constr_pu05_HIC		0.8071
y_{12}	constr_pb05_ChestD		0.9444	y_{38}	constr_pu05_Head_IP_min		0.8982
y_{13}	constr_pb05_ChestG		0.8499	y_{39}	constr_pu05_NeckFzMax		0.6210
y_{14}	constr_pb05_Chest_IP_min		0.9742	y_{40}	constr_pu05_NeckFzMin		0.8027
y_{15}	constr_pb05_FemurL	Yes	0.9746	y_{41}	constr_pu05_Nij		0.6222
y_{16}	constr_pb05_FemurR		0.8697	y_{42}	constr_pu50_ChestD	Yes	0.7207
y_{17}	constr_pb05_HIC	Yes	0.6117	y_{43}	constr_pu50_ChestG		0.6611
y_{18}	constr _pb05_Head_IP_min		0.8216	y_{44}	constr_pu50_Chest_IP_min	Yes	0.9039
y_{19}	constr_pb05_NeckFzMax		0.8955	y_{45}	constr_pu50_FemurL		0.9349
y_{20}	constr_pb05_NeckFzMin		0.8434	y_{46}	constr_pu50_FemurR		0.9364
y_{21}	constr_pb05_Nij		0.8368	y_{47}	constr_pu50_HIC		0.6182
y_{22}	constr_pb50_ChestD	Yes	0.7593	y_{48}	constr_pu50_Head_IP_min		0.9075
y_{23}	constr_pb50_ChestG		0.5197	y_{49}	constr_pu50_NeckFzMax		0.6073
y_{24}	constr_pb50_Chest_IP_min		0.7118	y_{50}	constr_pu50_NeckFzMin		0.7084
y_{25}	constr_pb50_FemurL		0.8613	y_{51}	constr_pu50_Nij		0.6923
y_{26}	constr_pb50_FemurR		0.8428				

rest of the points (1,171) were considered for testing. Although the training data set has all the levels, it is still a small number of points for training comparing the number of points for testing therefore, 722 additional points removed from the testing data were added, having a total of 1,000 points for training and 449 for testing. Other data combination was made by simply switching the training and testing data sets that is, considering 1,249 points for training and 200 for testing. Three different designs were created for each of the data sets considering 39 variables and the number of training

Table 5.16. Variance inflation factor of the regressors of the automotive vehicle case.

Variable	VIF	Variable	VIF
x_{1a}	1.08363	x_{18}	1.21475
x_{1b}	1.12087	x_{19}	1.35574
x_{1c}	1.59217	x_{20}	1.42988
x_2	1.10652	x_{21}	1.09646
x_3	1.38955	x_{22a}	1.52556
x_4	1.13043	x_{22c}	1.81054
x_5	1.18911	x_{23}	1.11020
x_6	1.28648	x_{24}	1.12729
x_7	1.11912	x_{25}	1.08684
x_8	1.11431	x_{26}	1.28156
x_9	1.11877	x_{27}	1.12230
x_{10}	1.31817	x_{28}	1.46902
x_{11}	1.12850	x_{29}	1.20569
x_{12}	1.09621	x_{30}	1.14706
x_{13}	1.33493	x_{31}	1.35961
x_{14}	1.15499	x_{32}	1.11227
x_{15}	1.12874	x_{33b}	1.45680
x_{16}	1.23094	x_{33c}	1.11986
x_{17}	1.08639		

Table 5.17. Minimum distance measures between design points.

Original data	Sobol	LHD	Uniform
0.1000	1.5517	1.6392	1.5255

points for each of them. Considering a design range of [0.0001, 0.9999]. Table 5.18 shows the minimum distance between points from the original data, Sobol design, LHD and Uniform design for each of the data sets.

From Table 5.18, it can be stated that the design for any of the new data sets is a good design since the minimum distance between points is very small in comparison to any of the other designs. This results as a limitation for this case of study since the statistical models created with any of the data sets will lack accuracy.

Table 5.18. Minimum distance measures between design points for different input data sets.

Data sets	Original data	Sobol	LHD	Uniform
278-1171	0.0833	1.6022	1.6040	1.5246
1000-449	0.0833	1.4209	1.4208	1.4007
1249-200	0.0000	1.4033	1.4311	1.3928

Continuing with the original data set (200 points for testing and 1,249 points for training) stepwise linear regression models and piecewise-linear MARS models were generated. The same procedure was followed for one of the described data sets (1,000 points for training and 449 points for testing) which considers the total number of variables (39). First the multiple linear regression models were created and the residual plots were analyzed to examine for possible existing curvature. In the same manner, stepwise linear regression models and piecewise-linear MARS models were generated. Tables 5.19 and 5.20 indicates the models that have curvature and the R-squared for the approximation models for the two different data sets.

The parameters used to create the piecewise-linear MARS approximations for the two data sets varies from model to model. However, all of them are restricted up to two-way interactions. Three-way interaction was also tested but did not improve the fit of models in most of the cases. Another limitation is that the number of knots T for the binary variables ($p = 2$) is only one as described in Section 3.2.3. For the continuous variables, the number of knots is constant across all of them, since it is set based on the minimum number of levels for any dimension, $T = p - 2$. In this case, $T = 1$ since the minimum number of levels is $p = 3$ (Table 5.14). To define the maximum number of basis function M_{max}, different models were tested based on a trial and error approach and the one selected was the one that provides the

Table 5.19. Stepwise linear and MARS approximation models for two different data sets (1).

ID	200 input points			1000 input points		
	Curvature	Stepwise Linear Regression	Piecewise Linear MARS	Curvature	Stepwise Linear Regression	Piecewise Linear MARS
y_1		0.7742			0.6998	
y_2		0.5681			0.6220	
y_3		0.3723			0.6332	
y_4	Yes	0.7538	0.7812	Yes	0.6584	0.7878
y_5	Yes	0.6457	0.9176	Yes	0.6795	0.8785
y_6	Yes	0.7015	0.9317	Yes	0.6666	0.8671
y_7		0.4728			0.1408	
y_8		0.7831			0.7162	
y_9	Yes	0.6227	0.7291		0.3695	
y_{10}		0.5542			0.3349	
y_{11}	Yes	0.7172	0.7063		0.3689	
y_{12}		0.9438			0.9436	
y_{13}		0.8488			0.5540	
y_{14}		0.9739			0.9646	
y_{15}	Yes	0.9741	0.9826		0.7116	
y_{16}		0.8661			0.4749	
y_{17}	Yes	0.6044	0.7976	Yes	0.7120	0.8794
y_{18}		0.8196			0.7176	
y_{19}		0.8937			0.6734	
y_{20}		0.8403			0.6129	
y_{21}		0.8348			0.5990	
y_{22}	Yes	0.7562	0.7767	Yes	0.8188	0.9107
y_{23}		0.5143			0.3915	
y_{24}		0.7088			0.6701	
y_{25}		0.8582		Yes	0.7986	0.9242
y_{26}		0.8418		Yes	0.8224	0.9363

best R-squared for the testing data. For the data set of 200 points, the maximum number of selected basis functions was 10 and the minimum was 6. The R-squared for the training data varies from 0.7063 to 0.9826, however the R-squared for testing data was very poor, the maximum R-squared was 0.5046 while the minimum was

Table 5.20. Stepwise linear and MARS approximation models for two different data sets (2).

	200 input points			1000 input points		
ID	Curvature	Stepwise Linear Regression	Piecewise Linear MARS	Curvature	Stepwise Linear Regression	Piecewise Linear MARS
y_{27}		0.7203			0.3034	
y_{28}		0.8483			0.7924	
y_{29}		0.5822		Yes	0.6054	0.7682
y_{30}		0.6535			0.3006	
y_{31}		0.6287			0.4150	
y_{32}		0.4373			0.2444	
y_{33}		0.8376			0.2403	
y_{34}		0.8795			0.7635	
y_{35}		0.5538			0.2615	
y_{36}		0.5607			0.4517	
y_{37}		0.8060			0.4740	
y_{38}		0.8965			0.7948	
y_{39}		0.6175			0.3980	
y_{40}		0.8003			0.5198	
y_{41}		0.6192			0.4593	
y_{42}	Yes	0.7175	0.8185		0.3967	
y_{43}		0.6566			0.3738	
y_{44}	Yes	0.9020	0.9389		0.7546	
y_{45}		0.9337		Yes	0.8748	0.9424
y_{46}		0.9358		Yes	0.8605	0.9383
y_{47}		0.6129			0.5179	
y_{48}		0.9051			0.8665	
y_{49}		0.6023			0.3980	
y_{50}		0.7027			0.3261	
y_{51}		0.6842			0.4314	

-1.4110. A negative occurs when the model prediction is worse than just using the average response as the prediction. This can happen when the data for training and testing do not come from the same population. If the training data set is poorly designed, then it will not be representative of the predictor space, and it may not represent the same population as the testing data set. By looking at the formula

$R_2 = 1 - \frac{SSE}{SST}$ if the sum of squares of residuals (SSE) is larger than the total sum of squares (SST) the ratio would be greater than one, yielding a negative R-squared. For the data set of 1,000 points, the maximum number of selected basis functions was 52 and the minimum was 15. The R-squared for the training data varies from 0.7682 to 0.9424. For the testing data set the R-squared varies from 0.6561 to 0.9147. This perhaps is an indication that the 449 testing points are representative of the 1,000 training points. However, comparing the R-squared for training of the stepwise linear regression models, there are more models with lower R-squared (R-sq < 0.70) using the 1,000 points than using the 200 points. Additionally, some of the models seem to behave differently according to the input points, as is the case of the model for variable y_9 which showed curvature in the first data set but not in the second data set.

5.2.0.2 Conclusions

Since it was proved that the input data come from a poor design, the results for the statistical model might not be reliable. However, for the purpose of optimization, the models that are recommended are the ones generated from the 200 input points, since the quality of fit of the training data set shows better results than the quality of fit of the models generated from the 1,000 input points, although this data set is missing two levels (37 variables instead of the 39).

5.3 Air Quality Problem

Air quality refers to the state of the air around us. The quality of the air is measured by the concentration of different pollutants during certain periods of time. The United States Environmental Protection Agency (EPA) considers ozone as one of these pollutants. Ozone is a molecule formed by three oxygen atoms (O_3). It is formed naturally in the ozone layer from atmospheric oxygen by electrical discharge to ultraviolet radiation, and it is also produced on the lower atmosphere by the photochemical reaction of certain pollutants. The protective ozone or good ozone is the one formed in the upper atmosphere and the harmful ozone or the bad ozone is the one formed in the lower atmosphere. Ozone pollution can be defined then as the high concentration of ozone at the ground level. Ground-level ozone is highly corrosive and can lead to serious health damage; it is also a risk factor for the environment. It is created by a complex series of reactions involving nitrogen oxides ($NO_x = NO+NO_2$) and volatile organic compounds ($VOCs$) in the presence of high temperatures and sunlight. The primary sources of NOx are power plants, automobiles and industries. $VOCs$ are also emitted by cars and industries but they have also a natural source, which is the vegetation. Ozone controls then seek to reduce $VOCs$ and NO_x.

The following is a quote from Yang et al. ([2]): "Complex 3-D air quality photochemical models (e.g., Urban Airshed Model, U.S. EPA 1990; Comprehensive Air quality Model, http://www.camx.com/) have been developed to simulate air pollution emissions, chemical reactions, and atmospheric transport, in order to predict ozone concentrations and help government decision-makers evaluate control strategies". Other studies on ozone pollution control include Seinfeld and Kyan [97], Trijonis [98] and Loughlin et al. [99]. More recently, Yang et al. [100], [2] and Sule et al. [101]. Yang, et al. suggested a Decision-Making Framework (DMF) that searches for dynamic and targeted control policies to reduce ozone pollution. To quote Yang et al. ([2]): "This approach requires a lower total reduction of emissions than current control strategies based on trial and error approach typically employed by state government decisions-makers".

The next section describes in more detail the DMF strategy and focuses on the importance of applying sequential MARS algorithms in such application.

5.3.1 Sequential MARS

The objective of the DMF is to help decision-makers evaluate critical factors for reducing ozone emissions in order to achieve the EPA ozone standard in a more cost-effectively manner than other approaches. In order for the DMF to search for emission reduction control strategies, a comprehensive 3-D air chemistry photochemical model is used. The framework utilizes a rigorous continuous-state stochastic dynamic programming (SDP) formulation (Chen [23]) and employs mining and metamodeling tools to develop a computationally-efficient representation of the relevant ozone air chemistry. The approach consists on different steps. It first identifies the potential SDP state variables, decision variables and time stages, that are then used to define the desired SDP cost objectives and constraints. The framework then identifies the key state and decision variables using the comprehensive air chemistry photochemical model, and it ultimately estimates the SDP state transition equations. This task happens in the element called Atmospheric Chemistry Module. And it is considered to be the most critical task for the DMF. Finally, the SDP method brings all components together and solves for an optimal reduction policy.

The DMF prototype concentrated on an Atlanta case study for the ozone episode (eight-hour average exceeding 0.08 parts per million (ppm)) during the time period from July 29 to August 1, 1987, which is considered to be one of the worst on record to date. It is important to mention that Atlanta is NO_x limited, which means that targeting $VOCs$ emissions is not effective. Hence, the focus on this study is on NO_x emissions. It is assumed that the cost function for reduction of emissions at each source is convex and has a monotonic increment. The SDP can be formulated following the Equation 5.1 in Section 5.1.1. The objective for the ozone pollution SDP is to minimize the cost of avoiding ozone episodes. The state variables x_i at a given time potentially include concentrations of ozone, NOx, and VOC at various spatial locations across the metropolitan Atlanta region. Similarly, the decision variables u_t potentially include emissions of NOx and VOC at various locations and times over the course of the day. The transition functions $f_t(\cdot)$ represent the ozone pollution air chemistry. A photochemical air quality model such as the Urban Airshed Model (UAM, EPA, 1990) can be used to calculate transitions; however this simulation is computationally intensive. Yang et al. [100] studied mining and metamodeling tools to incorporate a more efficient approach into the SDP optimization. The Atmospheric Chemistry Module consists of three main phases, initialization, mining and metamodeling. The objective is to find the relationship between emissions and ozone from the comprehensive air chemistry photochemical model output by means of statistical methods from data mining and computer experiments. The details of the Atmospheric Chemistry Module are given in [100]. Basically, the challenge is to do a dimension reduction and to create appropriate experimental designs to fit the chosen approximation method, i.e. metamodel. After successfully reducing the dimensions of the SDP system which initially consists of more than 500 variables, the number of required state variables for stages 1 through 4 is 17, 25, 23 and 19 respectively. Yang et al. [100], [2] employed an experimental design based on a low-discrepancy sequence to discretize the continuous SDP state space and multivariate adaptive regression splines (MARS) approximations of the continuous SDP future value function. Yang et al. used sets of 2,000 points from a Sobol sequence. MARS models were fitted using ASR and robust MARS (Tsai and Chen [21]). The maximum number of basis functions was $M_{max} = 2,000$, the number of eligible knots was set to $T = 35$ and it was restricted up to two-way interactions. Backward MARS algorithm was not used for generating the models and the quintic function (Equations 3.6 and 3.7) for smoothing the approximation model was employed. Table 5.21 from [2] shows the number of selected basis functions by the MARS models and the computational times for each stage.

Table 5.21. MARS results for approximating the SDP future value functions and corresponding computational times.

Stage	Number of basis functions selected by MARS	Running time (hh:mm:ss)
1	1,754	41:08:28
2	1,089	13:44:33
3	120	0:18:55
4	290	0:31:03

Overall, the results obtained from the DMF suggest that it is much more cost-effective to target reductions by time and location, and that dynamic controls might be beneficial.

As it is demonstrated in this air quality case study, the usage of design and analysis of computer experiments (DACE) plays an important role when trying to analyze this type of complex systems. However, the computational effort and time is still a concern for researches. The goal of using sequential MARS approaches is to reduce this computational time and efficiently represent the system. The rest of this section will be focused on the application of sequential MARS algorithms on the data extracted from the last stage of the SDP problem. The name for the 19 state variables for the last stage is in Table 5.22 along with their minimum and maximum values. From the Sobol sequence, the first consecutive 1,800 points were considered as a training data and the rest of the points, 200 were considered as testing data. The different five algorithms described in Chapter 4 were applied and fitted to the data. All the algorithms started with 50 initial data points for the first iteration. They were then incremented by 50 more points at each iteration. The initial maximum number of basis functions was $M_{max} = 40$ and they incremented by ten more basis function at each iteration. The number of eligible knots was $T = 10$ except for sequential MARS 5 that used $T = 1$ for the first iteration. The input parameters change through the iterations based on the sequential MARS algorithm used. To evaluate the performance of these methods, 15 iterations (from 0 to 14) were performed for each of them. To measure the quality of fit, the mean squared error (MSE) was calculated for each iteration for each method. Note that the response variable was first standardized. The general and constant input parameters used for the approximations are shown in Table 5.23. Tables 5.24 to 5.28 show the results of the air quality application for each sequential MARS approach. The tables contain the changing input parameters for all the iterations that is, the number of input data points N, the maximum number of basis functions M_{max} and the number of eligible knots T. They also have the cumulative number of candidate knots, the number of selected basis functions and the MSE for each iteration.

5.3.1.1 Conclusions

Figure 5.14 shows the mean squared error for the different methods for every iteration. Sequential MARS 1 fits the data from scratch at every iteration, so every time it defines a new set of eligible knots. Sequential MARS 2 simply refits the β_m coefficients at every iteration, therefore the number of basis functions and knots are unchanged. Sequential MARS 3 generates a new MARS approximation using the residuals as a response, consequently the method defines also a set of eligible knots for every iteration. Sequential MARS 4 adds the functions generated from each

Table 5.22. State variables for the last stage of the air quality SDP problem.

ID	Variable	Minimum	Maximum
1	sq1_4p1	92.17	44,611.59
2	sq2_4p1	145.61	72,804.38
3	sq3_3p1	1,274.60	634,756.10
4	sq3_4p1	234.35	117,176.80
5	sq1_4p2	77.54	38,767.54
6	sq2_4p2	112.09	56,046.38
7	sq3_2p2	446.19	223,096.30
8	sq3_4p2	186.57	93,189.53
9	sq1_4p3	79.59	39,796.90
10	sq3_2p3	454.66	227,330.00
11	sq3_3p3	997.87	498,932.70
12	sq4_2p3	81.16	40,578.75
13	pt5p2	91.55	45,774.93
14	pt3p3	302.28	151,142.30
15	pt4p3	312.12	156,057.50
16	pt6p3	103.57	51,733.89
17	cyM3p3	0.01	0.12
18	skM3p3	0.01	0.12
19	tkM3p3	0.01	0.12

Table 5.23. General and constant input parameters for sequential MARS approaches.

Interactions	3
ASR	2
ASR difference	0.002
Robust	0
Robust tolerance	N/A
Original backward algorithm	0
Convex	0
Threshold	0.00
Smooth	0

iteration, but the eligible knots for previous iterations are used for any future iteration. Sequential MARS 5 maintains the existing basis functions and allows interaction

Table 5.24. Sequential MARS 1 applied to the last stage of the SDP air quality problem.

	Data	Input parameters		Candidate	Selected	
Iteration	points	$Mmax$	T	knots	basis functions	MSE
0	50	40	10	10	34	0.8997
1	100	50	10	10	40	0.8952
2	150	60	10	10	46	1.1495
3	200	70	10	10	51	0.3599
4	250	80	10	10	57	0.5902
5	300	90	10	10	62	0.3630
6	350	100	10	10	68	1.3843
7	400	110	10	10	73	0.2263
8	450	120	10	10	79	0.3194
9	500	130	10	10	84	0.1468
10	550	140	10	10	88	0.1030
11	600	150	10	10	93	0.1516
12	650	160	10	10	99	0.0544
13	700	170	10	10	104	0.1290
14	750	180	10	10	109	0.0896

effects with them, along with new main and interaction effects. It also follows the same structure for the knots as sequential MARS 4. Any previous eligible set of knots is used in the next iteration.

From the figure, it can be observed that sequential MARS 1 shows more instability through the iterations however, the MSE for the last iteration is very low (0.0896). For sequential MARS 2, the MSE result stabilizes at an earlier iteration, however the method cannot reach the accuracy level of sequential MARS 1. This implies that by increasing the model complexity, a better approximation model can be generated. Sequential MARS 3 and sequential MARS 4 in theory should be performing similarly, since they are based on a boosting approach, which guarantees a more robust approximation. Both methods show stability with few data points, but neither of the methods can reach the accuracy level of sequential MARS 1 at iteration

Table 5.25. Sequential MARS 2 applied to the last stage of the SDP air quality problem.

Iteration	Data points	Input parameters $Mmax$	Input parameters T	Candidate knots	Selected basis functions	MSE
0	50	40	10	10	34	0.8997
1	100	-	-	-	-	0.5734
2	150	-	-	-	-	0.5479
3	200	-	-	-	-	0.5619
4	250	-	-	-	-	0.4989
5	300	-	-	-	-	0.4905
6	350	-	-	-	-	0.4733
7	400	-	-	-	-	0.4350
8	450	-	-	-	-	0.4349
9	500	-	-	-	-	0.4153
10	550	-	-	-	-	0.4190
11	600	-	-	-	-	0.4164
12	650	-	-	-	-	0.4125
13	700	-	-	-	-	0.4178
14	750	-	-	-	-	0.4291

14. And between these two approaches, sequential MARS 4 seems to perform better than sequential MARS 3 since the MSE values are lower in general. This may have occurred because of the number of candidate knots it had for selecting the final set. In this case, sequential MARS 4 had a total of 38 candidate knots while sequential MARS 3 had 150 candidate knots. Having many knots may not be beneficial. These results show that the number of candidate knots for each dimension is an important parameter that can affect the accuracy of the approximation function. The number of selected knots for the last approximation function (iteration 14) was 40 in sequential MARS 4 and 87 in sequential MARS 3. Sequential MARS 3 in this case had higher probability to select more knots since the method is based on the summation of small approximation functions and each of them had different eligible knots. Sequential MARS 1 and MARS 2 had only 10 candidate knots. The number of selected knots at

Table 5.26. Sequential MARS 3 applied to the last stage of the SDP air quality problem. Note: the cumulative number of candidate knots is the number of candidate knots from previous iterations plus the current number of eligible knots; however, to generate the approximation function at each iteration only 10 knots were candidates to choose from.

	Data	Input parameters		Candidate	Selected	
Iteration	points	$Mmax$	T	knots	basis functions	MSE
0	50	40	10	10	34	0.8997
1	100	50	10	20	44	0.7563
2	150	60	10	30	54	0.5628
3	200	70	10	40	64	0.5772
4	250	80	10	50	72	0.5133
5	300	90	10	60	81	0.4542
6	350	100	10	70	90	0.3898
7	400	110	10	80	100	0.3537
8	450	120	10	90	109	0.3507
9	500	130	10	100	119	0.3560
10	550	140	10	110	128	0.3246
11	600	150	10	120	137	0.2362
12	650	160	10	130	147	0.2118
13	700	170	10	140	157	0.2004
14	750	180	10	150	166	0.2047

iteration 14 in sequential MARS 1 was 72 while in sequential MARS 2 was only 19. Other results for sequential MARS 3 and sequential MARS 4 are shown in Table 5.29 using a smaller number of knots, a total of 15 candidate knots for sequential MARS 3 and a total of 10 candidate knots for sequential MARS 4. The number of data points N and maximum number of basis function M_{max} is the same as before. Both methods show comparable results, however sequential MARS 4 provides slightly better results using a smaller number of knots. In the case of sequential MARS 3, the results in Table 5.26 are slightly better. The cumulative number of candidate knots (15) is significantly smaller than the one previously used (150), however there is no flexibility for selecting the knots at each iteration since the eligible knots is $T = 1$. Other

Table 5.27. Sequential MARS 4 applied to the last stage of the SDP air quality problem.

Iteration	Data points	Input parameters $Mmax$	T	Candidate knots	Selected basis functions	MSE
0	50	40	10	10	34	0.8997
1	100	50	2	12	44	0.7698
2	150	60	2	14	54	0.4671
3	200	70	2	16	63	0.3482
4	250	80	2	18	73	0.2526
5	300	90	2	20	83	0.2426
6	350	100	2	22	93	0.2396
7	400	110	2	24	103	0.2300
8	450	120	2	26	113	0.2221
9	500	130	2	28	123	0.2194
10	550	140	2	30	133	0.1968
11	600	150	2	32	143	0.1933
12	650	160	2	34	153	0.1912
13	700	170	2	36	163	0.1967
14	750	180	2	38	173	0.2059

combination of parameters should be tested keeping a small cumulative number of knots but allowing more flexibility for the eligible knots at each iteration. Similarly, for sequential MARS 5, different runs were tested adding the flexibility to choose from more knots. Approach 1 used up to 80 candidate knots and the results were reasonable but not better than sequential MARS 1, 3 or 4 (Tables 5.24, 5.26 and 5.27. Conversely, approach 2 used only 10 candidate knots but these knots were the same through all the iterations, and the results did not improve. Table 5.30 shows these preliminary results for sequential MARS 5. The number of data points N and maximum number of basis function M_{max} are the same as before. Finally, the input parameters that mimicked sequential MARS 1 the best were the ones shown in Table 5.28, where one knot was added at each iteration until the set of candidate knots was 10. A total of 87 knots were selected in the final approximation (iteration 14).

Table 5.28. Sequential MARS 5 applied to the last stage of the SDP air quality problem.

	Data	Input parameters		Candidate	Selected	
Iteration	points	$Mmax$	T	knots	basis functions	MSE
0	50	40	1	1	36	1.2551
1	100	50	1	2	44	0.7725
2	150	60	1	3	54	0.5770
3	200	70	1	4	62	0.4280
4	250	80	1	5	71	0.1995
5	300	90	1	6	80	0.2157
6	350	100	1	7	90	0.2259
7	400	110	1	8	100	0.1860
8	450	120	1	9	110	0.1564
9	500	130	1	10	120	0.1303
10	550	140	0	10	130	0.1144
11	600	150	0	10	140	0.0954
12	650	160	0	10	150	0.0902
13	700	170	0	10	160	0.0822
14	750	180	0	10	170	0.0823

Sequential MARS 5 is probably the most similar method to sequential MARS 1, with the difference being that sequential MARS 5 is forced to have the previous selected basis functions, as well as the previous eligible knots. The results in Figure 5.14 show that the best result over all the methods for iteration 14 is the one obtained from sequential MARS 5 (0.0823) which is slightly better than sequential MARS 1. The MSE values for sequential MARS 5 also seem to be more stable.

Figure 5.15 shows the values of the future value function of the last stage of the SDP of the variables cyM3p3 (maximum ozone level at Conyers during period 3) and skM3p3 (maximum ozone level at South DeKalb during period 3), the rest of the variables were fixed using their median value (from Yang et al.). Similarly, Figure 5.16 illustrates the future value function of these two variables, except using the MARS approximation function generated by sequential MARS 1 at iteration 14.

Table 5.29. Other results for sequential MARS 3 and 4 applied to the last stage of the SDP air quality problem.

Sequential MARS 3				Sequential MARS 4			
T	Candidate knots	Selected basis functions	MSE	T	Candidate knots	Selected basis functions	MSE
1	1	36	1.2551	1	1	36	1.2551
1	2	46	0.5385	1	2	46	0.4961
1	3	56	0.4826	1	3	56	0.4569
1	4	65	0.3969	1	4	66	0.3247
1	5	75	0.4253	1	5	76	0.3013
1	6	85	0.3953	1	6	86	0.2807
1	7	95	0.3547	1	7	96	0.2622
1	8	105	0.3520	1	8	106	0.1998
1	9	115	0.3524	1	9	116	0.1865
1	10	125	0.3277	1	10	126	0.1831
1	11	135	0.3241	0	10	136	0.1882
1	12	145	0.3105	0	10	146	0.1879
1	13	155	0.2738	0	10	156	0.1832
1	14	165	0.2815	0	10	166	0.1650
1	15	169	0.2770	0	10	176	0.1622

It is observed that they are comparable, however, the presence of non-convexities in Figure 5.16 is visible.

The challenge with the sequential approaches is the selection of the input parameters. This will probably depend a lot on the application, but right now the way to define them is based on a trial and error strategy. Regardless of the possible difficulty for defining the best set of input parameters, this approach seems to be very promising for complex systems that are computationally intensive to solve. For the last stage of the air quality SDP problem, the sequential methods show reasonable results by only using less than half of the total input data points (750 out of 1,800), reaching an MSE of 0.0823. For reference, if considering all the data input points $N = 1,800$, a maximum number of basis functions of $M_{max} = 390$, a total of eligible knots of $T = 35$ and

Table 5.30. Preliminary results for sequential MARS 5 applied to the last stage of the SDP air quality problem.

Approach 1				Approach 2			
T	Candidate knots	Selected basis functions	MSE	T	Candidate knots	Selected basis functions	MSE
10	10	34	0.8997	10	10	34	0.8997
5	15	44	28.0180	0	10	42	0.7288
5	20	53	1.0999	0	10	55	0.8806
5	25	63	2.0984	0	10	62	2.5116
5	30	71	0.6477	0	10	71	2.2843
5	35	80	0.5763	0	10	89	1.3469
5	40	90	0.4674	0	10	99	1.2413
5	45	100	0.5267	0	10	109	1.1361
5	50	110	0.6056	0	10	119	1.3689
5	55	120	0.3070	0	10	129	1.3554
5	60	130	0.2898	0	10	139	1.1006
5	65	140	0.3383	0	10	149	1.1358
5	70	150	0.2656	0	10	159	1.0219
5	75	160	0.2427	0	10	169	1.0111
5	80	170	0.2218	0	10	179	1.0349

considering also the general input parameters shown in Table 5.23, the MSE reached is of 0.0346 selecting a total of 214 basis functions including main effects, two and three-way interactions effects. All of the MARS sequential approaches have two and three-way interaction effects in their last MARS approximation. Table 5.31 shows the cumulative computational times for the sequential approaches executed on a Quad 3.00-GHz 8GB RAM Dell Precision Workstation. The sequential MARS approach that takes the longest cumulative time is obviously sequential MARS 1 since it has to build all the basis functions in every iteration. On the other hand, the approach that takes the least amount of time is sequential MARS 2. Sequential MARS 3 and 4 are very competitive taking only few seconds and sequential MARS 5 takes significantly more time than sequential MARS 2, 3 and 4, but much less than sequential MARS 1.

The MARS approximation using all the points and the input parameters mentioned above takes 04:32:54, however if instead of $T = 35$, the number of knots is $T = 10$, this time is reduced to 01:45:55 reaching an MSE=0.0445. In any instance, sequential approaches are much more efficient in terms of the computational time and can still generate a good approximation fit.

As stated before, it is assumed that the future value function of the air quality SDP problem is convex, however the MARS approximations used to evaluate the performance of the sequential MARS approaches are nonconvex. Convexity can be selected as the input setting for any of the approaches, however a final check should be added to guarantee convexity. This will consist of performing the BIPR algorithm at the end of each MARS approximation.

Table 5.31. Cumulative computational times (hr:min:sec) for sequential MARS approaches applied to the last stage of the SDP air quality problem.

Iteration	Sequential MARS 1	Sequential MARS 2	Sequential MARS 3	Sequential MARS 4	Sequential MARS 5
1	0:00:01	0:00:01	0:00:01	0:00:01	0:00:00
2	0:00:03	0:00:01	0:00:01	0:00:01	0:00:00
3	0:00:08	0:00:01	0:00:01	0:00:01	0:00:02
4	0:00:15	0:00:01	0:00:01	0:00:01	0:00:05
5	0:00:29	0:00:01	0:00:02	0:00:02	0:00:13
6	0:00:52	0:00:01	0:00:02	0:00:02	0:00:25
7	0:01:29	0:00:01	0:00:02	0:00:03	0:00:47
8	0:02:18	0:00:01	0:00:02	0:00:03	0:01:19
9	0:03:23	0:00:01	0:00:03	0:00:04	0:02:05
10	0:04:52	0:00:01	0:00:03	0:00:05	0:03:09
11	0:07:10	0:00:01	0:00:03	0:00:06	0:04:29
12	0:10:08	0:00:01	0:00:04	0:00:08	0:06:12
13	0:14:15	0:00:01	0:00:04	0:00:09	0:08:20
14	0:18:21	0:00:01	0:00:05	0:00:11	0:10:55
15	0:24:50	0:00:01	0:00:06	0:00:13	0:14:00

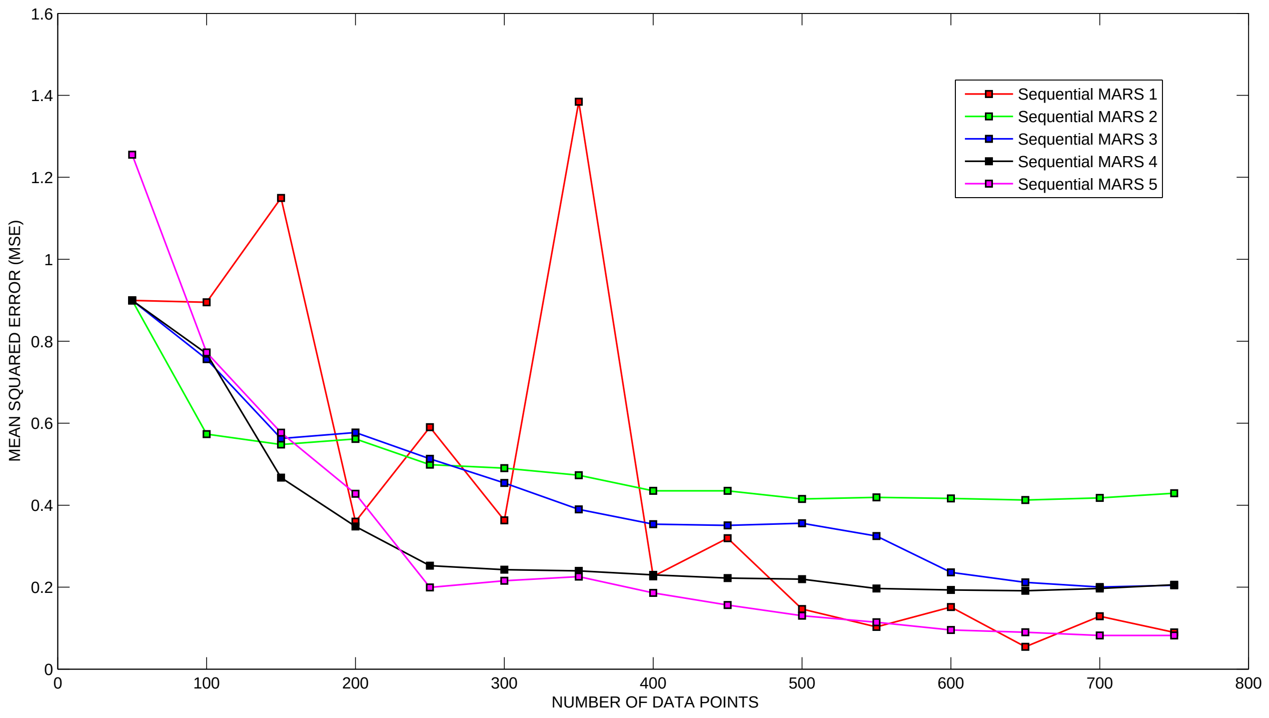


Figure 5.14. MSE results from the different sequential MARS algorithms.

5.3.2 Convex MARS Applied in Dynamic Programming

In this section, the same air quality application described in Section 5.3.1 is studied with the purpose of applying convex MARS. As it is already known, the future value function is in theory convex, therefore it is favorable to use an approximation method that can handle convexity. Original MARS cannot guarantee convexity due to the structure of the interaction terms in the basis functions. As can be observed from Figure 5.16, the estimated future value function shows some non-convexities, and this complicates the optimization process.

The data extracted from the last SDP stage are slightly different from Yang's work since Ariyajunya [3] refined the framework. Particularly, he focused on the

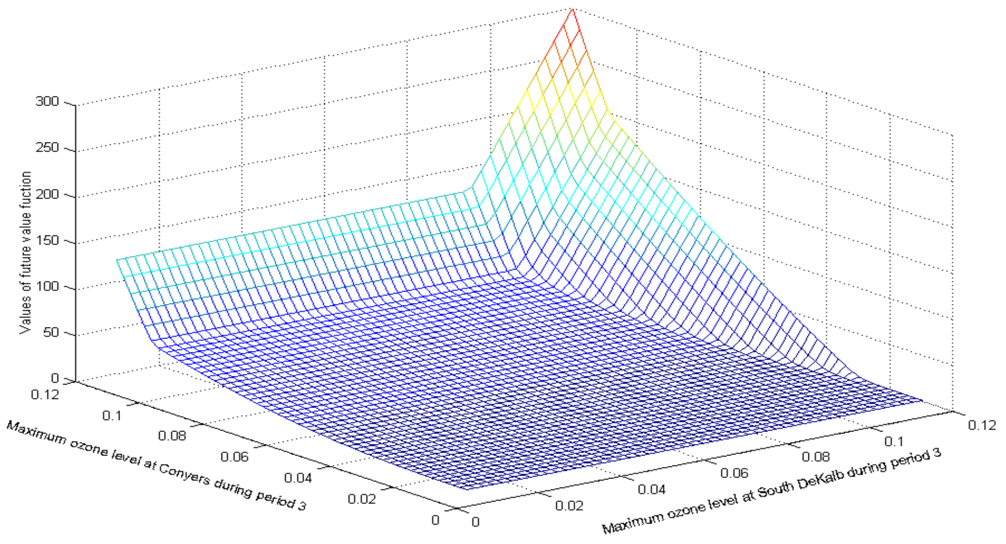


Figure 5.15. Last period of the air quality SDP optimal value function from Yang et al. [2].

modeling phase for the transition function. The transition functions represent how the state of the system evolves from the time of the current stage to time of the next stage. From the Equation 5.1 the transition function is:

$$x_{t+1} = f_t(x_t, u_t, \epsilon). \tag{5.2}$$

The equation represents the dynamics of the system as a function of the state variables (x_t) at time t, the decision variables u_t and uncertainty ϵ. In this application the transition function is unknown. Ariyajunya [3] proposed different types of state transition metamodels for the Atlanta case.

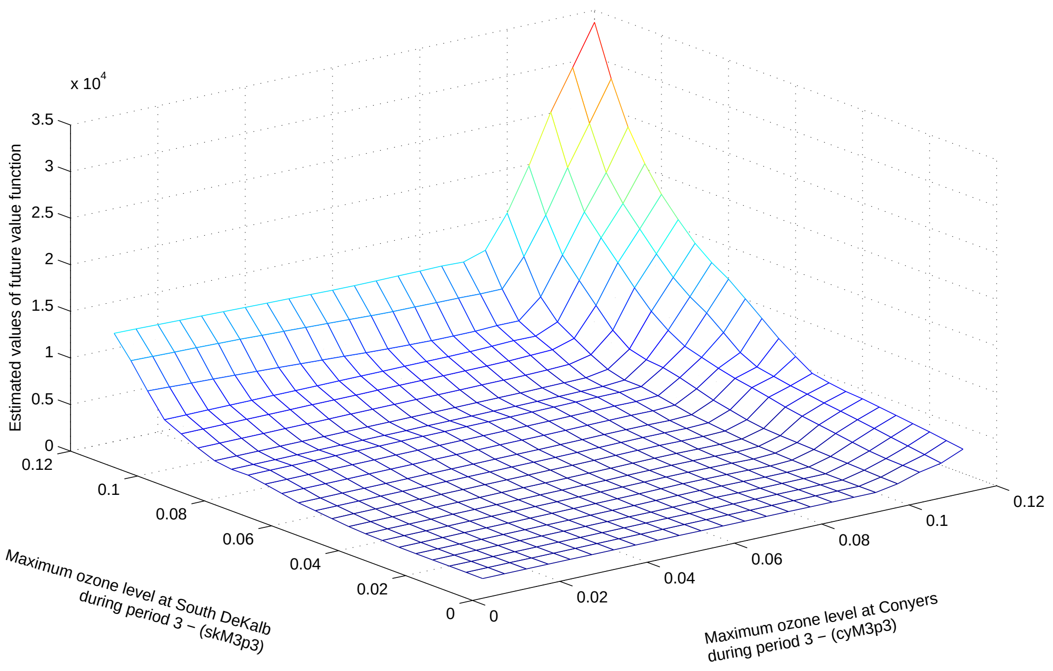


Figure 5.16. Last period of the air quality SDP optimal solution value function estimated by sequential MARS 1 (iteration 14) (Table 5.24).

As mentioned before, the DACE approach based SDP uses experimental design and statistical models to approximate the value function. An ideal experimental design (orthogonal) will not be appropriate to use when the variables are correlated and the Atlanta case is known to have a multicollinear state space. Therefore, one of the types of metamodels studied by Ariyajunya addresses the multicollinearity between ozone concentrations at different times. Multicollinearity is measured by the variance inflation factor (VIF) from a regression model. Ariyajunya considers low multicollinearity if VIF is less than 4 and high multicollinearity if VIF is greater than 10. He tested the case with high VIF's which does not address multicollinearity allowing high VIF metamodels and the case with low VIF's which address multicollinearity by carefully crafting regression models to obtain low VIF metamodels.

The procedure consists basically on identifying those models with high VIF values (VIF > 4) and correct them by removing some of the high correlated predictors, refitting and re-evaluating the models. Additionally, if predictors are not significant based on their p-value, models should be also corrected by using stepwise regression to select only statistically significant predictors (significance level of 0.05). The SDP method first used a backward solution to approximate the future value function starting from the last stage and moving backward until all stages are solved, and a forward SDP re-optimization in a "real-time" simulation to re-solve for the optimal decisions. Ariyajunya followed the work of Yang et al. and utilized also a low-discrepancy sequence by Sobol of 2,000 points to discretize the space. At each design point, a nonlinear programming method was used to solve for an optimal solution, and a commercial optimization Fortran library (NAG E04) was used as the optimization module for solving the SDP. At each stage, the solution of the backward SDP is the MARS approximation of the future value function. To address the potential for local optima, multiple starting points can be used for the optimization module, this increases the chance of getting close to the global optimal cost. See [3] for results when using multiple starting point for the optimal cost.

Table 5.32 from [3] shows the number of selected basis functions by the MARS models and the computational times for each stage. These results were generated using the low VIF metamodels approach for the state transition functions and the backward solution of the SDP only. It used two starting points (midpoint and lower bound) and an additional ten random points within the ranges for computational reasons. The negative MARS approximation values were truncated to zero since having a negative cost is not realistic. The common parameters for the MARS approximation used for each stage were $N = 2,000$, $M_{max} = 2,000$ and $T = 35$. It uses the ASR and is restricted to two-way interaction terms. Backward MARS algorithm was not used for generating the models and the quintic function (Equations 3.6 and 3.7) for smoothing the approximation model was employed.

Table 5.32. MARS results for approximating the SDP future value functions and corresponding computational times.

Stage	Number of basis functions selected by MARS	Running time (hh:mm:ss)
1	394	0:53:31
2	1,853	50:09:49
3	104	0:02:47
4	90	0:02:30

Using the low VIF metamodels tends to require lower emission reduction than using other metamodels, also the computational effort for solving the SDP is less. See [3] for more detailed results. This approach developed by Ariyajunya seems to be accurate for predicting the maximum ozone level, however since original MARS is used to approximate the future value function, the nonconvexity issue is still present.

Convex MARS has been applied to the last stage of the SDP. The strategy for defining the threshold (τ) in convex MARS is still under study. The approach described in Section 2.1.2 was tested, but the results were not very satisfactory. In particular, the BIPR algorithm removes too many basis functions that do not satisfy the convexity constraint. This can lead to a poor MARS model fit. The maximum absolute coefficient from the original MARS model is 111,225.4729. This original MARS model comes from Ariyajunya's SDP approach but using one starting point (midpoint). This MARS model has also 90 basis functions. Different percentages were tried but since the threshold was highly strict, the final approximation model ended up with very few basis functions. Table 5.33 shows the threshold used, the changing parameters, the number of basis functions in the forward algorithm and the final number of basis functions. It also indicates if the final approximation is convex or not. All the approximations were generated using a total number of eligible knots of $T = 35$ and up to two-way interactions.

Table 5.33. Results using different thresholds (τ) for modeling the last stage of the SDP air quality problem.

%	τ	M_{max}	ASR	ASR difference	Basis functions in forward algorithm	Selected basis function	Convex
1	1,113	2,000	2	0.0002	92	7	yes
		500	-	N/A	269	66	no
2	2,225	2,000	2	0.0001	100	13	yes
		500	-	N/A	269	67	no
3	3,337	2,000	2	0.0001	85	5	no
		500	-	N/A	269	68	no
5	5,562	500	-	N/A	269	65	no

As it is observed in Table 5.33 none of the values used for threshold provided a good approximation, since a lot of the basis functions were removed. Also, nonconvex approximations are possible in the convex MARS method. This happens because once the BIPR algorithm is performed, the β_m coefficients are refitted and a second pass on the BIRP is executed. If any of the sum of the coefficients for pair or unpaired selected basis functions does not satisfy the convexity restriction, then the final approximation is nonconvex.

Other different random, but much lower, values were tested. Finally a threshold value managed to work well. This threshold value range is between 182 and 240. Using $M_{max} = 2,000$ and ASR (ASR difference= 0.0002), the forward algorithm has a total of 64 basis functions, and the final model remains with 51 basis functions.

If using a threshold value $\tau = 181$, the number of basis functions in the forward algorithm was 64 and the number of basis functions selected in the final model was 8. Similarly, if the threshold value $\tau = 241$, the number of basis functions in the forward algorithm was 76, and the number of basis functions selected in the final model was 7. Another approach to determine the threshold was then derived. Taking into consideration also the original MARS function, now the average (μ) of the sum of the coefficients either pair or unpaired of only the univariate terms (main effects) is calculated along with its standard deviation (σ). Then the coefficient of variation is calculated ($\mathrm{CV} = \frac{\sigma}{\mu}$). A percentage is selected and then the threshold is calculated as follows, $\tau = \textit{percentage} \cdot \frac{\mu}{\mathrm{CV}}$). The criterion to select the percentage is not yet defined. But for this case, the taget is to find a percentage that provides the value of the best threshold previously obtained ($182 \leq \tau \leq 240$). The following represents the threshold calculation for the last stage SDP MARS approximation model:

$$\mu = 2,972.59,\ \sigma = 17,010.32,\ \mathrm{CV} = 5.72,$$

if percentage$=$ 0.351, then the threshold is $\tau = 182.33$, if percentage$=$ 0.463, then the threshold is $\tau = 240.51$. The percentage range for determining the threshold is then $0.351 \leq \% \leq 0.463$. Following the same procedure but using the median instead of the average, the calculations are as follows:

$$\textit{median} = 3,218.11,\ \sigma = 17,010.32,\ \frac{\sigma}{\textit{median}} = 5.29,$$

if percentage$=$ 0.30, then the threshold is $\tau = 182.65$, if percentage$=$ 0.395, then the threshold is $\tau = 240.48$. The percentage range for determining the threshold is then $0.30 \leq \% \leq 0.395$. This approach is probably more robust than the previous one tested (Section 2.1.2) since it involves a measure of central tendency and its dispersion.

In order to test the performance of this proposed approach, the same calculations were computed for the four- and nine-dimensional inventory forecasting problem described in Section 5.1. The target value for the threshold considered was $\tau = 30$ for the four-dimensional case and $\tau = 15$ for the nine dimensional case. These threshold values were the ones that yielded the best results in the original data (without noise) (see Table 5.5 in Section 5.1.1). Tables 5.34 and 5.35 show the results. The estimated threshold value ($\hat{\tau}$) is comparable with the target threshold value (τ). However, similarly as in the air quality case, the approach using the median seems to be closer to the target value. More cases should be tested in order to define a rule to determine the percentage, but the objective is to have it nearly constant.

Table 5.34. Calculating the threshold value using the proposed approach based on the mean for the inventory forecasting case.

	τ	μ	σ	$\text{CV} = \frac{\sigma}{\mu}$	%	$\hat{\tau} = \% \cdot \frac{\mu}{\text{CV}}$
Four-dimensional	30	119.42	122.96	1.03	0.25	29.00
Nine-dimensional	15	108.82	130.06	1.20	0.25	22.76

Table 5.35. Calculating the threshold value using the proposed approach based on the median for the inventory forecasting case.

	τ	median	σ	$\text{CV} = \frac{\sigma}{\text{median}}$	%	$\hat{\tau} = \% \cdot \frac{\text{median}}{\text{CV}}$
Four-dimensional	30	122.16	122.96	1.01	0.30	36.41
Nine-dimensional	15	69.48	130.06	1.87	0.30	11.14

Figure 5.17 shows the values of the future value function of the last stage of the SDP of the variables cyM3p3 (maximum ozone level at Conyers during period 3) and skM3p3 (maximum ozone level at South DeKalb during period 3); the rest of the variables were fixed using their median value (from Ariyajunya). Similarly, Figure 5.18 illustrates the future value function of these two variables but using the

convex MARS approximation function. The approximation plot is comparable with the future value function. It is worth mentioning that the metamodels developed by Yang et al. have a slightly different set of state variables than the metamodels developed by Ariyajunya. The number of state variables for stages 1 through 4 is 16, 23, 21 and 19 respectively. The last stage which is the one studied here, has the same dimensions than before, and the variables used for generating the future value function plots are present in both metamodels (Yang et al. and Ariyajunya). However, Table 5.22 should not be considered here since the variables might be the same but the ranges are different.

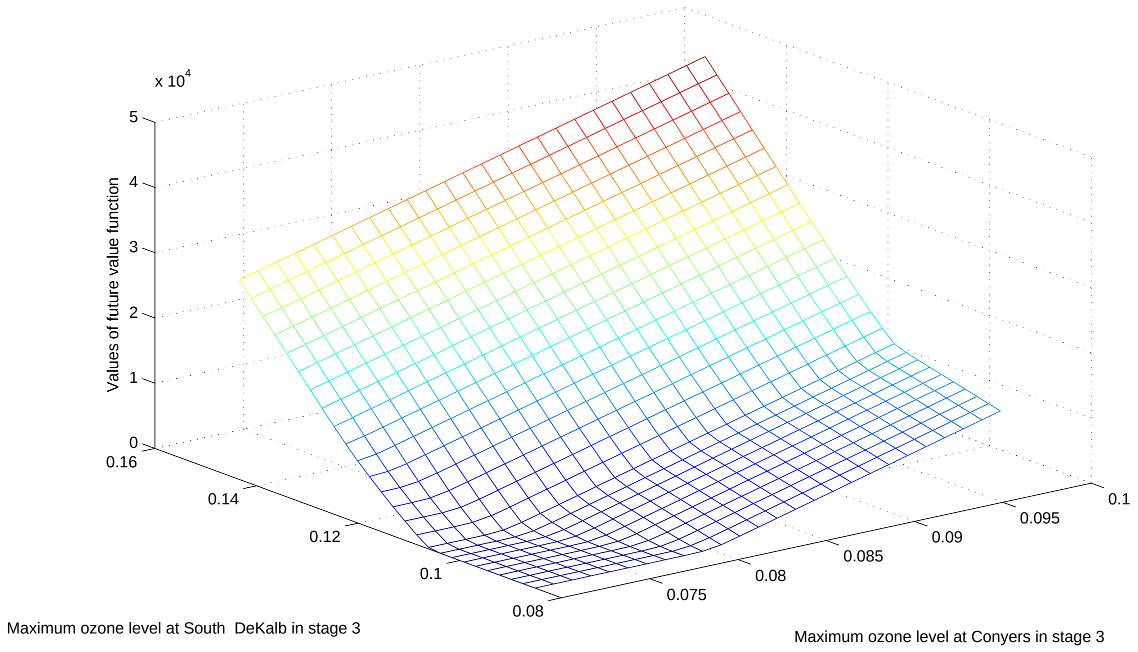


Figure 5.17. Last period of the air quality SDP optimal value function from Ariyajunya [3].

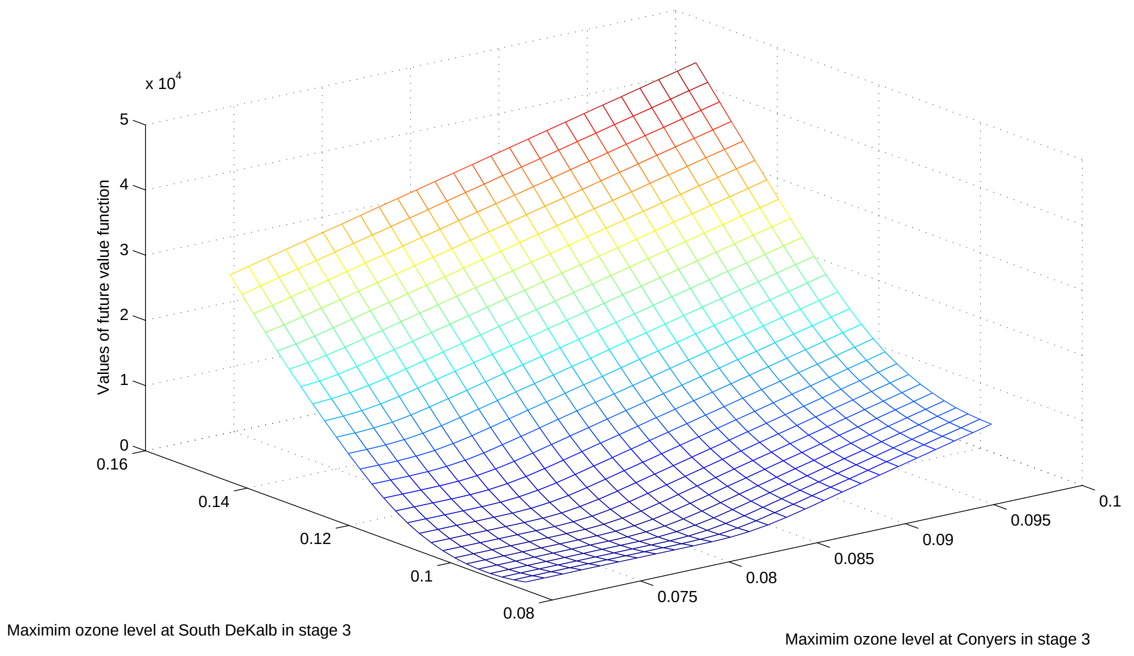


Figure 5.18. Last period of the air quality SDP optimal solution value function estimated by convex MARS.

5.3.2.1 Conclusions

Modeling the air quality value function with convex MARS is ideal since in theory the true function is convex. This will also facilitates the optimization process by being able to find a global optimal solution. However some difficulties have been faced in the attempt of doing this. One is finding the threshold that leads to a good approximation. This final approach seems to be promising but more tests should be performed. The other difficulty is that the approximations could be nonconvex as it was seen in Table 5.33; this might be happening because the data are nonconvex. A way to adapt convex MARS method to data that are nonconvex should be explored in the future. Another alternative is to model the future value function using the nonconvex piecewise linear version of MARS and optimize it using the proposed method by Martinez et al. [102] which guarantees global optimization.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

Multivariate adaptive regression splines (MARS) is a non-interpolating and flexible regression method that can provide a good approximation globally and can sufficiently capture the shape of any function. MARS structure has data-driven capability, and it excludes the non-important variables. Additionally, one of its most important properties, is that MARS is flexible in its structure, but it can still be restricted to satisfy certain structural properties.

The developed MARS variants have basically the same original search algorithm, except that the interaction basis functions are reconstructed by a linear transformation. Convex MARS variants differ from the nonconvex variants, since the selection of the basis functions is restricted to a nonnegative sum of coefficients. The primary objective is to provide more flexibility to the modeling process by enabling specification of properties of the approximation model based on the properties of the true function. And if later, this is desired to be optimized, a more appropriate optimization process can be selected based on the properties of the function approximation. There are basically four different variants of MARS, convex piecewise-linear, convex with the smoothing option, nonconvex piecewise-linear and nonconvex with the smoothing option. These last two have the additional option to select the original backward algorithm if desired. The MARS variants were tested using the inventory forecasting case study with four and nine dimensions. The value function for this case is known to be convex, therefore each of the MARS variants should be evaluated using more case studies.

MARS variants that can handle binary variables are those who require a piecewise-linear fit only, since the only possible values for the knots are -1 and 1 and the quintic function for smoothing routine requires also side knots. If desired, one could adapt the algorithm to enable a quintic fit for those functions with continuous variables and leave the piecewise-linear fit for those basis functions with at least one binary variable. Although this is possible to do, it is difficult to say if this will be beneficial for the approximation.

A limitation that can easily be addressed is the number of eligible knots for each variable. As for now, the algorithm has this number constant across the variables. The user defines the number of desired eligible knots, however if this number is greater than the minimum number of levels of the input variables, then it is set to a lower value. This limits the flexibility of selecting more knots. However if the number of knots is set individually this problem can disappear.

Sequential MARS enables the capability of updating the approximation function once new data are received. This approach can be used in dynamic programming applications where the objective or other functions need to be estimated. The advantage is to reduce the computational effort by using fewer data points than a complete experimental design.

There are five different alternatives to choose which are:

- Sequential MARS 1 - fit a MARS function from scratch at each iteration,
- Sequential MARS 2 - update the estimated model coefficients at each iteration,
- Sequential MARS 3 - build on an existing MARS function: Sum of MARS approximations based on residuals,
- Sequential MARS 4 - build on an existing MARS function: Sum of MARS approximations,

- Sequential MARS 5 - build on an existing MARS function: One MARS approximation.

Sequential MARS 1, 3 and 5 should not have any issue performing the different MARS variants. However, sequential MARS 4 and 5 cannot perform the MARS variants that require the smoothing routine. These methods keep the previous eligible knots for any future iteration and have the option to select more or just keep the initial set. The algorithm described in this dissertation to choose new knots is specifically for the center knots, therefore the quintic fit could not be made since it requires the side knots. This, however, is something to consider as future work since it should be possible to also select new side knots and maintain the existing ones.

Sequential MARS was tested in the data extracted from the last stage of the SDP air quality problem showing satisfactory results. However, in this case study, in theory, the future value function is convex, but the approximation functions within the sequential approaches are nonconvex. The task of checking and maintaining convexity requires performing the BIPR algorithm and removing those basis functions that violate the convexity constraint. The ability to do this within the sequential approaches is an issue of future work. For convex MARS in the SDP air quality, one big issue is a way to determine the threshold on the sum of coefficients. An alternative way was proposed that seems to be promising, however more tests should be performed. Additionally, convex MARS for nonconvex data should be explored.

Another concern about the sequential approaches is the setting of the input parameters. This is application-dependant but general guidance still needs to be defined. Studies of more case studies could reveal better guidance.

Finally, the sequential approaches can be applied in optimization problems, such as SDP or surrogate optimization, where a function needs to be estimated. Therefore it will be ideal to embed the different options into an optimization routine. Although the proposed sequential framework enables combinations of the sequential approaches (see Figure 4.1), this has not been implemented. Future work can develop an appropriate approach to enable combinations, especially within optimization problems.

APPENDIX A

MARS ALGORITHMS

Figures A.1 and A.2 show the original forward and backward algorithms for MARS.

```
$B_1(\mathbf{x}) \leftarrow 1$; $M \leftarrow 2$
Loop until $M > M_{\max}$: $\text{lof}^* \leftarrow \infty$
  For $m = 1$ to $M - 1$ do:
    For $v \notin \{v(k,m) | 1 \leq k \leq K_m\}$
      For $t \in \{x_{vj} | B_m(\mathbf{x}_j) > 0\}$
        $g \leftarrow \sum_{i=1}^{M-1} a_i B_i(\mathbf{x}) + a_M B_m(\mathbf{x})[+(x_v - t)]_+ + a_{M+1} B_m(\mathbf{x})[-(x_v - t)]_+$
        $\text{lof} \leftarrow \min_{a_1, \ldots, a_{M+1}} \text{LOF}(g)$
        if $\text{lof} < \text{lof}^*$, then $\text{lof}^* \leftarrow \text{lof}$; $m^* \leftarrow m$; $v^* \leftarrow v$; $t^* \leftarrow t$ end if
      end for
    end for
  end for
  $B_M(\mathbf{x}) \leftarrow B_{m^*}(\mathbf{x})[+(x_{v^*} - t^*)]_+$
  $B_{M+1}(\mathbf{x}) \leftarrow B_{m^*}(\mathbf{x})[-(x_{v^*} - t^*)]_+$
  $M \leftarrow M + 2$
end loop
end algorithm
```

Figure A.1. Original MARS Forward Algorithm.

```
$J^* = \{1, 2, \ldots, M_{\max}\}$; $K^* \leftarrow J^*$
lof* $\leftarrow \min_{\{a_j | j \in J^*\}} \mathrm{LOF}(\sum_{j \in J^*} a_j B_j(\mathbf{x}))$
For $M = M_{\max}$ to 2 do: $b \leftarrow \infty$; $L \leftarrow K^*$
   For $m = 2$ to $M$ do: $K \leftarrow L - \{m\}$
      lof $\leftarrow \min_{\{a_k | k \in K\}} \mathrm{LOF}(\sum_{k \in K} a_k B_k(\mathbf{x}))$
      if lof $< b$, then $b \leftarrow$ lof; $K^* \leftarrow K$ end if
      if lof $<$ lof*, then lof* $\leftarrow$ lof; $J^* \leftarrow K$ end if
   end for
end for
end algorithm
```

Figure A.2. Original MARS Backward Algorithm.

Algorithm A.1 Convex MARS Interaction Transformation Algorithm (CIT)

```
z = ($\phi_m$ / (1$-\phi_m$ * $k_{L_m,m}$) * ($x_{v(L_m,m)} - k_{L_m,m}$)).
for all ($l = 1, 2, \ldots, L_m - 1$) do
  if $s_{l,m}$==1 then
    z += ($x_{v(l,m)} - k_{l,m}$)/(1$-k_{l,m}$).
  else if $s_{l,m}$==$-1$ then
    z $-$= ($x_{v(l,m)} - k_{l,m}$)/(1+$k_{l,m}$).
  end if
end for
return z.
```

Algorithm A.2 Convex MARS Forward Coefficient Restriction Algorithm (FCR)

```
Initialize $M = 1$, maxIA = maximum # input variables in an interaction.
while ($m < M_{\max}$) do
  LOF* = $\infty$.
  for all $m = 0, \ldots, M-1$ do
    if basis function $m$ involves fewer than maxIA input variables then
      for all $v = 1$ to $n$ do
        if $v \notin$ basis function $m$ then
          for all $k = 1$ to $K$ do
            for all candidate non-zero basis functions do
              if (coefficient $> 0$ or threshold from an unpaired basis function) $\cup$ (sum
              of coefficients $> 0$ or threshold from a pair of basis functions) then
                Calculate lack-of-fit LOF.
                if LOF < LOF* then
                  LOF* = LOF. Save $m*, v*, k*, \phi_m*$.
                end if
              end if nonnegative
            end for candidate basis functions
          end for $k$
        end if $v$
      end for $v$
    end if
  end for $m$
  Add basis functions $(m*, v*, k*, \phi_m*)$, $M$+=2.
  Orthonormalize new basis functions.
end while
```

Algorithm A.3 Convex MARS backward Iteration of Pruning and Refitting Algorithm (BIPR)

```
Initialize the full set of $m$ basis functions.
while not convex do
  for all basis functions $\subseteq$ current set $(i = m, m-1, \ldots, 1)$ do
    if negative coefficient $\cap$ unpaired then
      Drop $i$-th basis function.
      $m = m - 1$.
    else if negative sum of coefficients for a pair then
      Drop one of the pair of basis functions ($i$-th and $i+1$-th ).
      $m = m - 1$.
    else if negative coefficients for each of a pair then
      Drop the pair of basis functions ($i$-th and $i+1$-th ).
      $m = m - 2$.
    end if
    Refit convex MARS model if any basis functions have been dropped.
  end for
end while
```

REFERENCES

[1] D. T. Shih, "Convex versions of multivariate adaptive regression splines and implementations for complex optimization problems," Ph.D. dissertation, University of Texas at Arlington, 2006.

[2] Z. Yang, V. C. P. Chen, M. E. Chang, M. L. Sattler, and A. Wen, "A decision-making framework for ozone pollution control," *Operations Research*, vol. 57(2), pp. 484–498, 2009.

[3] A. Bancha, "Adaptive dynamic programming for high-dimensional multicollinear state spaces." Ph.D. dissertation, University of Texas at Arlington, 2012.

[4] V. C. P. Chen, K.-L. Tsui, R. R. Barton, and J. K. Allen, *A Review of Design and Modeling in Computer Experiments*, ser. In Handbook of Statistics (C. R. Rao and Ravi Khattree, eds.). NY: Elsevier Science, 2003, vol. 22, pp. 231–261.

[5] C. Vorwerg, G. Socher, T. Fuhr, G. Sagerer, and G. Rickheit, "Projective relations for 3d space: Computational model, application, and psychological evaluation," in *National Conference on Artificial Intelligence*, 1997, pp. 159–164.

[6] X. Wei, L. Wu, S. Hou, and Y. Li, "Aircraft take-off and landing performance intelligent computation model and applications," in *Proceeding on the 6th World Congress on Intelligent Control and Automation*, 2006.

[7] D. Drignei and D. E. Popescu, "Fast regression surrogates for computer models with time-dependent outputs," *Journal of Statistical Computation and Simulation*, pp. 1–10, 2012.

[8] M. J. Bayarri, J. O. Berger, J. Cafeo, G. Garcia-Donato, F. Liu, J. Palomo, R. J. Parthasarathy, R. Paulo, J. Sacks, and D. Walsh, "Computer model validation with functional output," *Annals of Statistics*, vol. 35, pp. 1874–1906, 2007.

[9] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments (with discussion)," *Statistical Science*, vol. 4, pp. 409–423, 1989.

[10] R. R. Barton, "Metamodels for simulation input-output relations," *Proceedings of the 1991 Winter Simulation Conference (J. J. Swain, D. Goldsman,et al., eds.)*, pp. 289–299, 1991.

[11] M. D. Morris, T. J. Mitchell, and D. Ylvisaker, "Bayesian design and analysis of computer experiments: Use of derivatives in surface prediction," *Technometrics*, vol. 35, pp. 243–255, 1993.

[12] C.-H. Tu and R. R. Barton, "Production yield estimation by the metamodel method with a boundary-focused experiment design," *Proceedings of the ASME Design Engineering Technical Conferences, DETC97*, 1997.

[13] R. R. Barton, "Simulation metamodels," *Proceedings of the 1991 Winter Simulation Conference (J. J. Swain, D. Goldsman,et al., eds.)*, vol. 1, pp. 167–174, 1998.

[14] F.-Y. Wang and S. Tang, "A framework for artificial transportation systems: from computer simulations to computational experiments," in *International Conference on Intelligent Transportation*, 2005.

[15] M. J. Alvarez, N. Gil-Negrete, L. Ilzarbe, M. Tanco, E. Viles, and A. Asensio, "A computer experiment application to the design and optimization of a capacitive accelerometer," *Applied Stochastic Models in Business and Industry*, vol. 25, pp. 151–162, 2009.

[16] I. M. Sobol, "The distribution of points in a cube and the approximate evaluation of integrals." *U. S. S. R. Computational Mathematics and Mathematical Physics*, vol. 7, pp. 86–112, 1967.

[17] J. M. Hammersley, "Monte carlo methods for solving multivariable problems." *Annals of the New York Academy of Sciences*, vol. 86, pp. 844–874, 1960.

[18] J. H. Friedman, "Multivariate adaptive regression splines (with discussion)," *Annals of Statistics*, vol. 19, pp. 1–141, 1991.

[19] V. C. P. Chen, D. Ruppert, and C. A. Shoemaker, "Applying experimental design and regression splines to high-dimensional continuous-state stochastic dynamic programming," *Operations Research*, vol. 47, pp. 38–53, 1999.

[20] S. Bakin, M. Hegland, and M. R. Osborne, "Parallel mars algorithm based on b-splines," *Computational Statistics*, vol. 15, pp. 463–484, 2000.

[21] J. C. C. Tsai and V. C. P. Chen, "Flexible and robust implementations of multivariate adaptive regression splines within a wastewater treatment stochastic dynamic program," *Quality and Reliability Engineering International*, vol. 21, pp. 689–699, 2005.

[22] G. W. Weber, I. Batmaz, G. Kksal, P. Taylan, and F. Yerlikaya-zkurt, "Cmars: a new contribution to nonparametric regression with multivariate adaptive regression splines supported by continuous optimization," *Inverse Problems in Science and Engineering*, vol. 20:3, pp. 371–400, 2012.

[23] V. C. P. Chen, "Application of mars and orthogonal arrays to inventory forecasting stochastic dynamic programs," *Computational Statistics and Data Analysis*, vol. 30, pp. 317–341, 1999.

[24] V. C. P. Chen, D. Günter, and E. L. Johnson, "Solving for an optimal airline yield management policy via statistical learning," *Journal of the Royal Statistical Society, Series C 52*, vol. 1, pp. 1–12, 2003.

[25] C. Cervellera, V. C. P. Chen, and A. Wen, “Neural network and regression spline value function approximations for stochastic dynamic programming,” *Computers and Operations Research*, vol. 34, pp. 70–90, 2006.

[26] S. Siddappa, “Statistical modeling approach to airline revenue management with overbooking,” Ph.D. dissertation, University of Texas at Arlington, 2006.

[27] V. Pilla, “Robust airline fleet assignment,” Ph.D. dissertation, University of Texas at Arlington, 2006.

[28] V. C. P. Chen, “Applying experimental design and regression splines to high-dimensional continuous-state stochastic dynamic programming,” Ph.D. dissertation, School of Operations Research and Industrial Engineering,Cornell University, Ithaca, New York, U.S.A., 1993.

[29] H. Fan, “Sequential frameworks for statistics-based value function representation in approximate dynamic programming,” Ph.D. dissertation, University of Texas at Arlington, 2008.

[30] R. Jin, W. Chen, and A. Sudjianto, “An efficient algorithm for constructing optimal design of computer experiments,” *Journal of Statistical Planning and Inferences*, vol. 134(1), pp. 268–287, 2005.

[31] J. Friedman and W. Stuetzle, “Projection pursuit regression,” *Journal of the American Statistical Association*, vol. 76, pp. 817–823, 1981.

[32] G. Atkenson, A. Moore, and S. Schaal, “Locally weighted learning,” *Artificial Intelligence Review*, vol. 11(1/5), pp. 11–73, 1997.

[33] S. Weiss and N. Indurkhya, “Rule-based regression,” *In Proceedings of the 13th International Joint Conference on Artificial Intelligence*, vol. b, pp. 1072–1078, 1993.

[34] W. S. and I. N., “Rule-based machine learning methods for functional prediction,” *Artificial Intelligence Research*, vol. 3, pp. 383–403, 1995.

[35] C. Stone, “Additive regression and other nonparametric models,” *Ann. Statist.*, vol. 13, pp. 689–705, 1985.

[36] G. Wahba, *Spline Models for Observational Data.* Philadelphia: SIAM, 1990.

[37] T. Hastie and R. Tibshirani, *Generalized additive models.* London, UK: Chapman and Hall, 1990.

[38] P. M. Kuhnert and R. McClure, “Combining non-parametric models with logistic regression: an application to motor vehicle injury data,” *Computational Statistics and Data Analysis*, vol. 34, pp. 371–386, 2000.

[39] D. C. Psichogios, R. D. De Veaux, and L. H. Ungar, “Non-parametric system identification: A comparison of mars and neural networks,” *American Control Conference*, pp. 1436–1441, 1992.

[40] E. N. Ben-Ari and D. M. Steinberg, “Modeling data from computer experiments: An empirical comparison of kriging with mars and projection pursuit regression,” *Quality Engineering*, vol. 19, pp. 327–338, 2007.

[41] C. Yazici, F. Yerlikaya-zkurt, and D. Batmaz, “A computational approach to nonparametric regression: Bootstrapping the cmars method.” *CFE-ERCIM 2011: 4th International Conference of the ERCIM (European Research Consortium for Informatics and Mathematics) Working Group on Computing and Statistics, London, UK.*, 2011.

[42] A. Özmen, G.-W. Weber, D. Batmaz, and E. Kropat, “Robustification of cmars with different scenarios under polyhedral uncertainty set.” *Communications in Nonlinear Science and Numerical Simulation (CNSNS): Nonlinear, Fractional and Complex*, vol. 16, pp. 4780–4787, 2011.

[43] E. Kartal-Ko, “An algorithm for the forward step of adaptive regression splines via mapping approach,” Ph.D. dissertation, Middle East Technical University, Ankara, Turkey, 2012.

[44] P. A. W. Lewis and J. G. Stevens, “Nonlinear modeling of time series using multivariate adaptive regression splines (mars),” *Journal of The American Statistical Association*, vol. 86, pp. 864–877, 1991.

[45] J. Deichmann, A. Eshghi, D. Haughton, S. Sayek, and N. Teebagy, “Application of multiple adaptive regression splines (mars) in direct response modeling,” *Journal of Interactive Marketing*, vol. 16, pp. 15–27, 2002.

[46] S.-M. Chou, T.-S. Lee, Y. E. Shao, and I.-F. Chen, “Mining the breast cancer pattern using artificial neural networks and multivariate adaptive regression splines,” *Expert Systems With Applications*, vol. 27, pp. 133–142, 2004.

[47] T.-S. Lee and I.-F. Chen, “A two-stage hybrid credit scoring model using artificial neural networks and multivariate adaptive regression splines,” *Expert Systems With Applications*, vol. 28, pp. 743–752, 2005.

[48] T.-S. Lee, C.-C. Chiu, Y.-C. Chou, and C.-J. Lu, “Mining the customer credit using classification and regression tree and multivariate adaptive regression splines,” *Computational Statistics and Data Analysis*, vol. 50, pp. 1113–1130, 2006.

[49] S. Crino and D. E. Brown, “Global optimization with multivariate adaptive regression splines,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 37, pp. 333–340, 2007.

[50] V. L. Pilla, J. M. Rosenberger, V. Chen, N. Engsuwan, and S. Siddappa, “A multivariate adaptive regression splines cutting plane approach for solving a two-stage stochastic programming fleet assignment model,” 2012.

[51] L. A. Hannah and D. B. Dunson, “Multivariate convex regression with adaptive partitioning,” *ArXiv e-prints*, May 2011.

[52] G. E. P. Box and K. B. Wilson, “On the experimental attainment of optimum conditions,” *Journal of the Royal Statistical Society*, vol. Series B 13, pp. 1–45, 1951.

[53] S. M. Stigler, “Gergonne’s 1815 paper on the design and analysis of polynomial regression experiments,” *Historia Mathematica*, vol. 1(4), pp. 431–439, 1974.

[54] G. E. P. Box and N. R. Draper, “A basis for the selection of a response surface design,” *Journal of the American Statistical Association*, vol. 54, pp. 622–654, 1959.

[55] ——, *Empirical Model Building and Response Surfaces.* New York, NY: John Wiley and Sons, 1987.

[56] R. H. Myers and D. C. Montgomery, *Response Surface Methodology: Process and Product Optimization Using Designed Experiments.* New York, NY: John Wiley and Sons, 1995.

[57] A. A. Giunta, “Aircraft multidisciplinary design optimization using design of experimetns theory and response surface modeling methods,” Ph.D. dissertation, Virginia Polytechnic Institute and State University, 1997.

[58] M. Kendall, *A Course in Multivariate Analysis.* London, UK: Griffin, 1957.

[59] H. Hotelling, “The relations of the newer multivariate statistical methods to factor analysis,” *British Journal of Statistical Psychology*, vol. 10(2), pp. 69–79, 1957.

[60] I. T. Jolliffe, “A note on the use of principal components in regression,” *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 31(3), pp. 300–303, 1982.

[61] B. L., J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees.* Belmont, CA.: Wadsworth, 1984.

[62] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning.* New York: Springer-Verlag, 2009.

[63] I. Uysal and H. A. Güvenir, "An overview of regression techniques for knowledge discovery," *Knowledge Engineering Review*, vol. 14(4), pp. 319–340, 1999.

[64] A. Mayer, "Projection pursuit mixture density estimation," *IEEE Transactions on Signal Processing*, vol. 53, pp. 4376–4383, 2005.

[65] ——, *Projection Pursuit Fitting Gaussian Mixture Models*, 2002.

[66] L. Zhang, P. Wang, Y. Dong, C. Liu, C. Shan, and L. Gong, "Quality evaluation for artillery based on projection pursuit method," 2012.

[67] C. Fu, L. Han, Q. Li, X. Wang, X. Liu, and P. Li, "Network anomaly detection based on projection pursuit regression," in *IEEE International Symposium on Parallel and Distributed Processing with Applications Workshops*, 2011.

[68] H. Du, Z. Hu, A. Bazzoli, and Y. Zhang, "Prediction of inhibitory activity of epidermal growth factor receptor inhibitors using grid search-projection pursuit regression method," *Plos One*, vol. 6, 2011.

[69] H. White, "Learning in neural networks: a statistical perspective," *Neural Computation*, vol. 1, pp. 425–464, 1989.

[70] C. M. Bishop, *Neural Networks for Pattern Recognition.* New York: Oxford University Press, 1995.

[71] E. A. Nadaraya, "On estimating regression," *Theory of Probability and its Applications*, 1964.

[72] G. S. Watson, "Smooth regression analysis," *Sankhya: The Indian Journal of Statistics*, 1964.

[73] C. Cervellera, D. Maccio, and M. M., "Additive regression and other nonparametric models," *Neural Networks*, vol. 23 (7), pp. 917–925, 2010.

[74] L. S. Dharmasena, P. Zeephongsekul, and C. L. Jayasinghe, “Software reliability growth models based on local polynomial modeling with kernel smoothing,” in *International Symposium on Software Reliability Engineering*, 2011, pp. 220–229.

[75] T. Mühlenstädt and S. Kuhnt, “Kernel interpolation,” *Computational Statistics and Data Analysis*, 2011.

[76] G. Matheron, “Principles of geostatistics,” *Economic Geology*, 1963.

[77] T. J. Santner, B. J. Williams, and W. I. Notz, *The Design and Analysis of Computer Experiments.* New York: Springer Verlag.

[78] T. Mühlenstädt and S. Kuhnt, “Comparing different interpolation methods on two dimensional test functions,” 2009.

[79] R. Li and A. Sudjianto, “Analysis of computer experiments using penalized likelihood in gaussian kriging models,” *Technometrics*, 2005.

[80] Y. Xiong, W. Chen, D. Apley, and X. Ding, “A non-stationary covariance-based krigin method for metamodeling in engineering design,” *International Journal for Numerical Methods in Engineering*, 2007.

[81] L. Hartman and O. Hoessjer, “Fast kriging for large data sets with gaussian markov random fields,” *Computational Statistics and Data Analysis*, 2008.

[82] M. J. D. Powell, “Radial basis functions for multivariable interpolation: a review,” 1987.

[83] P. M. J. D., “Radial basis function methods for interpolation to functions of many variables,” 2001, pp. 2–24.

[84] M. Buhmann, *Radial Basis Functions.* Cambridge: Cambridge University Press, 2003.

[85] A. Conlin, J. Zhang, E. Martin, and A. Morris, “Statistical projection methods and artificial neural networks,” *Proceedings of the American Control Conference*, vol. 3, pp. 1852–1856, 1995.

[86] J. Muñoz and Felicísimo, “Comparison of statistical methods commonly used in predictive modeling,” *Journal of Vegetation Science*, vol. 15, pp. 285–292, 2004.

[87] J. R. Leathwick, J. Elith, and T. Hastie, “Comparative performance of generalized additive models and multivariate adaptive regression splines for statistical modelling of species distributions,” *Ecological Modelling*, vol. 199, pp. 188–196, 2006.

[88] R. Jin, C. Wei, and T. W. Simpson, “Comparative studies of metamodeling techniques under multiple modeling criteria,” *American Institute of Aeronautics and Astronautics*, vol. 312, pp. 996–6072, 2000.

[89] A. J. Booker, J. E. Dennis, F. Jr., S. P. D., V. D.B., Torczon, and M. W. Trosset, “A rigorous framework for optimization of expensive functions by surrogates,” *Structural Optimization*, vol. 17(1), pp. 1–13, 1999.

[90] J.-F. M. Barthelemy and R. T. Haftka, “Approximation concepts for optimum structural design - a review,” *Structural Optimization*, vol. 5, pp. 129–144, 1993.

[91] J. Sobieszczanski-Sobieski and R. T. Haftka, “Multidisciplinary aerospace design optimization: Survey of recent developments,” *Structural Optimization*, vol. 14, pp. 1–23, 1997.

[92] I. G. Osio and C. H. Amon, “An engineering design methodology with multistage bayesian surrogates and optimal sampling.” *Research in Engineering Design*, vol. 8, pp. 189–206, 1996.

[93] D. Huang, T. T. Allen, W. I. Notz, and N. Zeng, "Global optimization of stochastic blackbox systems via sequential kriging meta-models," *Journal of Global Optimization*, vol. 34, pp. 441–466, 2006.

[94] J. A. Christen and B. S. Sansó, "Advances in the sequential design of computer experiments based on active learning," *Communications in Statistics - Theory and Methods*, vol. 40(24), pp. 4467–4483, 2011.

[95] K. Crombecq, E. Laermans, and T. Dhaene, "Efficient space-filling and non-collapsing sequential design strategies for simulation-based modeling," *European Journal of Operational Research*, vol. 214(3), pp. 683–696, 2011.

[96] N. Quan, J. Yin, S. H. Ng, and L. H. Lee, "Simulation optimization via kriging: A sequential search using expected improvement with computing budget constraints," *Iie Transactions*, vol. just-accep, 2012.

[97] J. H. Seinfeld and C. P. Kyan, "Determination of optimal air pollution control strategies," *Socio-Economic Planning Science*, vol. 5, pp. 173–190, 1971.

[98] J. C. Trijonis, "Economic air pollution control model for los angeles county in 1975," *Environmental Science and Technology*, vol. 8(9), pp. 811–826, 1974.

[99] D. H. Loughlin, S. Ranjithan, J. W. Baugh, and E. D. Brill, "Application of genetic algorithms for the design of ozone control strategies," *Journal of the Air and Waste Management Association*, vol. 50, pp. 1050–1063, 2000.

[100] Z. Yang, V. C. P. Chen, M. E. Chang, T. E. Murphy, and J. C. C. Tsai, "Mining and modeling for a metropolitan atlanta ozone pollution decision-making framework," *IIE Transactions, Special Issue on Data Mining. COSMOS Technical Report 04-06.*, vol. 39, pp. 607–615, 2007.

[101] N. Sule, V. C. P. Chen, and M. L. Sattler, "A decision-making framework for assessing control strategies for ground-level ozone." *Atmospheric Environment*, vol. 45 (28), pp. 4996–5004, 2011.

[102] N. Martinez, D. Martinez, J. M. Rosenberger, and V. C. Chen, “Global optimization for a piecewise linear regression spline function.” *Proceedings of the 2011 Industrial Engineering Research Conference, Reno, NV.*, 2011.

BIOGRAPHICAL STATEMENT

Diana Martinez was born in Mexico city in 1983. She earned her B.S. in Industrial Engineering at the Instituto Tecnologico de Saltillo and worked in a plastic injection company as a Manufacturing and Project Engineer for over two years. In 2007 she was awarded a 60-month scholarship from a Mexican government organization to study at the University of Texas at Arlington. She finished her M.S. in Industrial Engineering in summer 2008 and then she joined the Ph.D. program in Industrial Engineering in the Center on Stochastic Modeling, Optimization, & Statistics (COSMOS). Additionally, she has been a Graduate Research Assistant for TMAC over the last five years and has been involved in different consulting projects for small and medium companies in the Dallas/Fort Worth metroplex. Projects mainly involve Lean manufacturing implementation, analysis of manufacturing operations, facility layouts and warehouse capacity analysis. Her current research interest is in the area of statistical modeling and data mining.