GLOBAL OPTIMIZATION OF NONCONVEX PIECEWISE LINEAR REGRESSION SPLINES

by

NADIA MARIA 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

I dedicate this dissertation to my parents for their unconditional love,

to whom I am eternally indebted.

ACKNOWLEDGEMENTS

I would like to express my most sincere gratitude and admiration to my advisor Dr. Jay M. Rosenberger. I have had the great opportunity to interact with him for years, and he is an exceptional mentor and instructor with outstanding knowledge. I thank him for all his guidance, dedication, patient, and support during my doctoral studies. It has also been a pleasure to have had Dr. Victoria Chen as a co-advisor; I thank her for introducing me to key researchers related to my area of interest and for the numerous advice and bright ideas provided during my research. Dr. Rosenberger and Dr. Chen are definitely two of the most brilliant people I have ever worked with. Thank you to my committee members Dr. Bill Corley and Dr. Bo Ping Wang for sharing their wisdom and encouraging me through these years; their support has been absolutely invaluable.

I would like to extend special thanks to Dr. Don Liles, who allowed me to be part of the story of this wonderful university and who believed in me. Thank you to my academic advisor Dr. Sheik Imrhan for guiding me properly to achieve my academic goals. I sincerely thank the entire faculty and staff members from the Industrial and Manufacturing Systems Engineering Department for all they have done to help me getting this achievement done; it has been a privilege sharing years of my life with these wonderful people.

I am very thankful to CONACYT (the Mexican National Council for Science and Technology), who granted me a scholarship to cover some of my educational expenses. Special thanks as well to Joe Crosswell and Drew Casani for giving me the

opportunity to work as a Graduate Research Assistant at TMAC and to the incredible and very knowledgeable co-workers from whom I have learned so much.

I owe a big thank you to one of the most talented and humble persons I ever known, Dr. Manuel Quevedo; he has undoubtedly been a key part of my personal and academic growth.

Many thanks to all my amazing friends, especially those from the Center on Stochastic, Modeling, Optimization and Statistics (COSMOS) and the System Engineering Research Center (SERC) for their endless support and encouragement to keep going. Thank you to my friend Giles DSilva for sharing his programming knowledge with me.

I am very grateful to my family who has given me strength to continue through all this time with their enormous love and support; to them I dedicate all my efforts. Thank you to my brother Macedonio Martinez for always believing in me. My most heartfelt thanks go to my twin sister and best friend Diana Martinez for all the work she has done to make my research possible; her love and for sharing her life with me, thank you for never letting me fall.

I have no words to express my gratitude to my biggest support in life, my lovely parents Maria Luisa Cepeda Tijerina and Macedonio Martinez Gonzalez; thank you for inspiring me every day. I just can't imagine my life without them.

Thank God for blessing me and letting me continue and enjoy this incredible life journey.

July 16, 2013

ABSTRACT

GLOBAL OPTIMIZATION OF NONCONVEX
PIECEWISE LINEAR REGRESSION SPLINES

NADIA MARIA MARTINEZ CEPEDA, Ph.D.

The University of Texas at Arlington, 2013

Supervising Professor: Jay M. Rosenberger

Global optimization has been applied to a wide variety of science and engineering design problems. Although numerous global optimization techniques have been developed and studied for decades, when used for complex systems such as the design of an aircraft or an automobile, the results are impractical or not completely satisfactory. One of the main drawbacks is the computational CPU time required to solve these problems. The majority of design problems require a significant number of experiments or simulations in order to find a globally best solution; however a single simulation can take between seconds and days to finish. One way to address the challenge of reducing these computation times is by building approximation models, known as *meta-models* or *surrogate models*. A surrogate model mimics the original model with a reduced number of simulations and has statistical properties that help develop patterns. With a surrogate model, an optimization method can then be applied to locate an optimum in an easier and faster manner; these methods are also well known as surrogate based global optimization methods. In recent years, a number of these types of approaches have been proposed. Some of the most repre-

sentatives meta-models are: polynomial regression, multivariate adaptive regression splines, radial basis functions, and kriging.

The surrogate model used in this study is multivariate adaptive regression splines (MARS). MARS is a flexible statistical modeling method, particularly useful for high-dimensional problems involving interactions and curvature. MARS terms are based on truncated linear functions, where interaction terms are formed as products. Hence, the univariate terms are piecewise linear, but the interaction terms are not. To enable use of fast mixed integer linear programming methods, the interaction terms of a MARS model are transformed to piecewise linear forms, which are more suitable for optimization.

Thus, the purpose of this research is to develop and demonstrate a deterministic global optimization method for a non-convex piecewise linear function generated by a modified version of MARS subject to constraints that include both linear regression models and piecewise linear MARS models. A new mixed integer linear programming (MILP) problem that can optimize the piecewise linear MARS model is formulated by introducing binary integer variables and a new set of inequalities to the model. The MILP problem is then solved by branch-and-bound, which is a widely used algorithm for solving various optimization problems. The MILP problem is adjusted to enable flexibility to handle both continuous and categorical variables.

This method is illustrated in an automotive crash safety system design case study for a major US automaker, where satisfactory computational results were obtained. The proposed method is compared with an evolutionary optimization algorithm.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Global optimization [1] is the branch of applied mathematics and numerical analysis that focuses on finding an optimal solution according to some criteria and fulfilling certain conditions. The solution is optimal if it finds the best (maximum or minimum) value of a given problem. For example, a business owner seeks to maximize profits while having costs as low as possible; or a building designer focused on reducing environmental impact, maximizes natural resources while minimizing energy and water usage. Optimization requires these types of problems to be represented in a mathematical form, where the decision parameters or alternatives are usually described by a vector of variables, some properties and /or restrictions involved in the problem are expressed by the constraint functions and the objective function, which is the output of the model that is being optimized.

There exists a wide variety of techniques to solve optimization problems, being the most prominent and practical one, linear programming method. The optimization can be obtained locally or globally i.e., the solution is locally optimal when it finds the best solution within a region of the feasible area, which is mainly defined by the limitations of the problem; and the solution is globally optimal when the best solution is found considering the entire feasible area and no better solution exists. Local optimization methods, which are generally less difficult in terms of computational complexity, guarantee a global solution under the assumption of convexity but many optimization problems involve non-convexity in their models i.e., they contain

[1] $http://en.wikipedia.org/wiki/Global-optimization$

multiple local optima, therefore the application of a global optimization technique is needed.

Global optimization has been applied to a wide variety of applied mathematics, medicine, molecular biology, finance, economics, networks and transportation, science, environmental, chemical, and other engineering design problems. Although numerous global optimization techniques have been developed and studied for decades, when used for complex systems such as the design of an aircraft or an automobile, the results are impractical or not completely satisfactory. One of the main disadvantages being the computational CPU time required to solve these problems.

The majority of design problems require a significant number of experiments or simulations in order to find a globally best solution. However a single simulation can take between seconds and days to finish. For example, to quote Wang and Shan [1]: "it is reported that it takes Ford Motor Company about 36-160 hrs to run one crash simulation [2]. For a two-variable optimization problem, assuming on average 50 iterations are needed by optimization and assuming each iteration needs one crash simulation, the total computation time would be 75 days to 11 months, which is unacceptable in practice." Although advances in technology have increased dramatically, such as the capacity and power of a computer, the previous example shows the complexity that many real world problems are still facing, especially when dealing with large scale problems. One way to address the challenge of reducing these computation times is by building approximation models, known as meta-models or surrogate models.

A surrogate model mimics the original model with a reduced number of simulations and has statistical properties that help develop patterns. With a surrogate model, an optimization method can then be applied to locate an optimum in an easier and faster manner. These methods are also well known as surrogate based

global optimization methods. In recent years, a number of these types of approaches have been proposed. Some of the most representatives meta-models are: response surfaces, kriging, radial basis functions, neural networks, and multivariate adaptive regression splines. The performance of these surrogate models on different problems varies due to the characteristics of each model. The surrogate model used in this study is multivariate adaptive regression splines (MARS).

MARS functions have been applied in numerical solutions to large-scale optimization problems, including dynamic programming [3–5]; revenue management [6]; and stochastic programming [7]. MARS is a flexible non-interpolating, non-parametric regression modeling method introduced by Jerome Friedman in 1991 [8]. MARS utilizes a forward-backward stepwise selection procedure in order to select the terms and prune the model respectively. It is particularly useful for problems that have a large number of design variables involving interactions and curvature. The parameters related with the MARS model and the number of basis functions are automatically determined by the data. MARS terms are based on truncated linear functions, where interaction terms are formed as products. Hence, the univariate terms are piecewise linear, but the interaction terms are not. Shih [9] modified the original MARS model to a piecewise linear function by transforming the multiple variables in an interaction term to a linear combination of the variables. The piecewise linear terms of the modified version of MARS are more suitable for optimization.

For optimizing the piecewise linear MARS model, a mixed integer linear programming (MILP) problem is needed. A mixed integer programming problem is the minimization or maximization of a piecewise linear function subject to linear constraints that deal with continuous variables and also with some variables restricted to integer values.

The most common algorithm used for solving integer programming problems is branch-and-bound, which was first proposed by A. H. Land and A. G. Doig in 1960 [10]. Branch-and-bound is a tree structure search method. It starts decomposing (branching) the feasible area of the original problem (root node) into smaller subproblems (child nodes); these subproblems are examined and solved obtaining an objective function value that represents a lower bound or upper bound (maximization or minimization problem, respectively) on the objective value of the original problem (bounding). It continues to partition the rest of the feasible area recursively, until a solution that satisfies the integer requirements is found and all the subproblems are examined and pruned. If the lower bound (upper bound) of a node is greater (less) than the best known feasible solution, then no solution exists in the feasible region of that node, and it can be discarded or pruned. If an optimal integer solution is found, the rest of the nodes can be pruned.

Thus, the contribution of this research is the development of a global optimization method of a non-convex piecewise linear function generated by a modified version of MARS subject to constraints that include both linear regression models and piecewise linear MARS models. A new mixed integer linear programming problem that can optimize the piecewise linear MARS model is formulated by introducing binary integer variables and a new set of inequalities and then solved by branch-and-bound. The MILP problem is adjusted to enable flexibility to handle both continuous and categorical variables.

The remainder of this research is organized as follows. Section 2 presents literature related and relevant to this study, which includes overviews of: global optimization methods; surrogate-based global optimization algorithms; studies related to piecewise linear functions and non-convex optimization; and the use of surrogate-based global optimization methods with a special interest in crashworthiness; followed

by the contribution of this research. Section 3, explains the original MARS and the modified piecewise linear version of MARS as the background of this research. It gives details about the proposed approach, which shows a new mixed integer linear programming formulation for optimizing two different problems: a single piecewise linear MARS function, and a piecewise linear MARS function subject to both piecewise linear MARS models and linear regression models. Section 4, shows results by demonstrating the method on a simple non-convex minimization inventory test problem and on an automotive crash safety system design for a major US automaker, which is the main focus of this research; a global optimization solution is obtained in both cases. Section 5 includes a comparison of the proposed method against the most common optimization technique used by researchers when combining the usage of surrogate models in regards to the second case study. This section also presents a validation procedure conducted to verify the solutions obtained. Finally, Section 6 presents directions for future work related to this research, which include the analysis of potential ways for improving the search of an optimal solution.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview of Global Optimization (GO) Methods

There exists an extensive variety of global optimization techniques [11, 12]; and the interest in developing new algorithms or improving existing ones keeps growing. A significant number of papers and books related to this topic is issued every year. The method proposed in this research attempts to find a global optima of a non-convex function, where the exact number of local minima or their locations are not known in advance. Since global optimization is essential in the proposed approach, this section presents a taxonomy of existing global optimization methods. Based on the literature, global optimization methods can be divided in two basic groups, deterministic and stochastic methods.

Deterministic methods also known as exact methods require a special mathematical model structure (such as concave, quadratic, bilinear, etc.) and guarantee to find a global optimum solution, assuring convergence. The basic idea of these methods is to disregard the regions where an optimal solution cannot be located. However, a drawback is that the computational time involved in using these methods usually increases exponentially as the model dimensionally grows. Review of such methods is provided in Floudas, 2000 [13] and Floudas and Pardalos, 2000 [14].

Some of the deterministic approaches are: Lipschitzian methods (Hansen et al., 1992 [15]), Relaxation (outer approximation) strategies (Hoffman, 1981 [16]; Duran and Grossmann, 1986 [17]; Horst et al., 1989 [18]), Branch-and-bound methods (Falk and Soland, 1969 [19]; McCormick, 1976 [20]; Al-Khayyal and Falk 1983 [21];

Gupta and Ravindran, 1985 [22]; Horst and Tuy, 1987 [23]; Al-Khayyal, 1990 [24]; Hansen, 1992 [25]; Ryoo and Sahinidis, 1996 [26]; Kearfott, 1996 [27]; Epperly and Swaney, 1996 [28]; Adjiman et al., 1998 [29]; Tawarmalani and Sahinidis, 1999 [30]), Cutting plane methods (Tuy, 1964 [31]; Hillestad and Jacobsen, 1980 [32]; Tuy et al., 1985 [33]), Decomposition methods (Tuy, 1987 [34]; Visweswaran and Floudas, 1993 [35]), Reformulation-linearization techniques (Sherali and Alameddine, 1992 [36]; Sherali and Tuncbilek, 1992 [37]), Interval methods (Hansen, 1979 [38]), Enumerative strategies (Horst and Tuy, 1996 [39]), Integral methods (Zheng and Zhuang, 1995 [40]), Primal-dual methods (Shor, 1990 [41]; Floudas and Visweswaran, 1990 [42]; Ben-Tal et al., 1994 [43]) and Deterministic heuristic DIRECT algorithm (Jones et al., 1993 [44]).

Stochastic methods also known as probabilistic methods include components that are random, such as the selection of the next step of computation and the generation of feasible trial points. The convergence to global optimality with the use of these methods is not guaranteed, however they can usually find solutions close to a global optimum in shorter computation times than deterministic methods. More information about these methods is provided in Guus et al., 1995 [45].

Some of the stochastic methods are: Random search and adaptive approaches (Brooks, 1958 [46]; Matyas, 1965 [47]; Rastrigin and Rubinstein, 1969 [48]; Zabinsky, 1998 [49]), Clustering algorithms (Rinnooy Kan and Timmer, 1987 [50]) and Multi-level single linkage methods (Rinnooy Kan and Timmer, 1987 [51]; Li and Chou, 1994 [52]).

Other techniques incorporate intelligent exploration and exploitation search procedures; these are well known heuristic strategies. Some of these methods are:

Approximate convex underestimation (Dill et al., 1997 [53]), Continuation methods (More and Wu, 1997 [54]), 'Globalized' extension of local search methods (Zhigl-

javsky, 1991 [55]; Pinter, 1996 [56]), Sequential improvement of local optima (Levy and Gomez, 1985 [57]), Hill Climbing (Rich and Knight, 1991 [58]), Simulated Annealing (Kirkpatrick et al., 1983 [59]), Tabu Search (Glover, 1977 [60]), Genetic Algorithms (Holland, 1975 [61]; Goldberg, 1989 [62]; Wu and Chow, 1995 [63]; Cheung et al., 1997 [64]), Evolution Strategies (Rechenberg, 1973 [65]), Ant Colony Optimization (Dorigo, 1992 [66]), Particle Swarm Optimization (Kennedy and Eberhart, 1995 [67]) and Differential Evolution (Storn and Price, 1997 [68]).

One of the inconveniences that global optimization techniques have is that they could either fail finding a feasible solution or they can get entrapped in local minima solutions that can highly vary from a global optima solution. Also, as was mentioned before, GO approaches require a significant number of evaluations of the objective and constraints functions for which the employment of traditional methods is not practical due to how expensive an evaluation may be in terms of computational times; therefore one way of decreasing these computation times is by building approximation models, known as surrogate models.

2.2 Overview of Surrogate-Based Global Optimization Algorithms

Surrogate modeling has been widely used in different disciplines such as statistics, mathematics, computer science, engineering and other areas of science. Surrogate models, meta-models, response surface models or emulators are intended to approximate the behavior of the true (original) function while being computationally less expensive.

In recent years, a significant number of approaches that use an approximation model as surrogate for optimization have been developed. This section aims to present the usage of the most representative surrogate models in optimization frameworks, such as: response surface models, kriging models, radial basis functions, neural net-

works and multivariate adaptive regression splines. It also presents recent related reviews and comparisons of several surrogate models.

In 1997, Guinta [69] developed the variable-complexity response surface modeling method to enable aircraft multisciplinary design optimization; Giunta et al. [70] showed a comparison of two approximation methods, quadratic polynomial models and kriging on several test problem, with one, five, and ten independent variables. Jones et al. [71] developed a method called Efficient Global Optimization (EGO) of expensive black box functions using kriging as the approximation model, which is especially good at modeling non-linear multimodal functions. In this method the next evaluation point is chosen to be the one that maximizes the expected improvement in the objective function value. Jin, et al. [72] presented a comparative study of four surrogate models: polynomial regression, multivariate adaptive regression splines, radial basis functions, and kriging; evaluating their performance based on different modeling criteria (accuracy, robustness, efficiency, transparency, and conceptual simplicity) in solving 14 test problems with different orders of non-linearity and problem scales. In 2001, Hosder et al. [73] successfully used polynomial response surface for multidisciplinary optimization of aircraft with up to about 30 variables. Jones [74] presented a taxonomy of response-surface based global optimization methods with a complete overview of different approaches; seven methods are discussed. Gutmann [75] introduced a global optimization method based on a general response surface technique; this method uses radial basis functions as interpolants and a measure of bumpiness is also available. The method was tested in a few numerical examples, showing favorable results in comparison to other global optimization methods. Bjorkman and Holmstrom [76] developed an improved radial basis function (RBF) algorithm and implemented it in Matlab. The RBF algorithm is based on the ideas presented by Gutmann [75], with some extension and further development. The efficiency of this

method is analyzed on a set of standard test optimization functions and on a real life industrial problem. In 2002, Emmerich et al. [77] presented the use of metamodels based on kriging techniques in the context of evolution strategies-based optimization algorithms. In 2003, Willmes et al. [78] showed the optimization performance of three well known test functions using evolution strategies assisted by meta-models such as kriging. Queipo, et al. [79] presented a discussion of the primary issues that occur in surrogate based analysis and optimization. Some design space sampling methods, different surrogate models, parametric and non-parametric approaches, model selection and validation, and surrogate based optimization were reviewed. Some of these issues were demonstrated for the aerospace industry. Wang and Shan [1] presented a review of different meta-modeling methods and their roles in support design optimization; some of the areas where these techniques can play a role involve: model approximation, design space exploration, problem formulation and the usage of these metamodels to solve various optimization needs. Regis and Shoemaker [80] introduced a stochastic response surface (SRS) method for the global optimization of expensive black box functions that utilizes a response surface model. A special case of SRS is also proposed. This method is called Metric SRS (MSRS), which utilizes a distance criterion when selecting the function evaluation points. A global optimization and a multistart local optimization version of MSRS were developed. Radial basis functions were used as the response surface models. Crino and Brown [81] proposed a global optimization procedure by combining multivariate adaptive regression splines with a response surface methodology. This approach was applied to seven test cases, all of them are low-dimensional examples. Forrester and Keane [82] studied and showed information about surrogate modeling methods, their use in optimization strategies, and their pros and cons of each of them. A guidance of the selection of the surrogate method encouraging a repetitive search and infill process is provided.

In 2010, Holena et al. [83] reported the use of neural networks as a surrogate model in evolutionary optimization of catalytic materials. Abramson et al. [84] introduced a new class of optimization problems that becomes less expensive as the solution is approached. It makes use of surrogates based on CPU times of previously evaluated points, rather than their function values. Gu et al. [85] proposed an intuitive strategy for global optimization, namely a hybrid and adaptive meta-modeling algorithm with a self-selecting mechanism for meta-model switching in the search process. Radial basis function, response surface method and kriging, three representative meta-models with unique capabilities, are the components of this algorithm.

Different factors should be taken into consideration when selecting a surrogate model. Although sometimes when there is little or no information about the behavior of the original function, it may not be easy. Some of the major factors to be considered are the problem size; the non-linearity of the function, if any; parameter settings established for each model; and whenever possible, the data collection method.

2.3 Piecewise Linear Functions and Non-Convex Optimization

This research considers the optimization of a non-convex piecewise linear MARS function modeled as mixed integer linear programming problem. Therefore some work related to these subjects is presented in the current section.

Some optimization problems involving non-convex piecewise linear functions can be found in applications such as water networks, heat exchanger networks, and distillation sequences [86] stationary gas network optimization [87], merge-in-transit distribution systems, including the integration of inventory and transportation decisions [88], among others.

Mixed integer programming models for non-convex piecewise linear functions have been broadly studied. They can be solved with algorithms such as the one

proposed by Keha et al. [89], a branch-and-cut algorithm without auxiliary binary variables for solving non-convex separable piecewise linear optimization problems that uses cuts and applies SOS2 branching. They can also be modeled as mixed integer programming (MIP) problems, following the work shown by Croxton et al. [90], where it was demonstrated that the linear programming relaxation of three textbook mixed-integer programming models for non-convex piecewise linear minimization problems are equivalent, each approximating the cost function with its lower convex envelope. The work presented by Keha et al. [91] showed that linear programming bounds of two well-known mixed-integer program formulations for piecewise linear functions, as well as the bounds of the continuous formulations are the same. Vielma, Keha, and Nemhauser [92] studied an extension of the branch-and-cut algorithm for solving linear problems with continuous separable piecewise linear cost functions developed by Keha et al. [91] in the case where the cost function is only lower semi-continuous. Using the global heuristic algorithm for solving a convex programming problem of maximizing a concave function on a convex domain was presented by Babayev and Bell [93]. The work previously cited focuses only on problems of separable functions. More recently Vielma et al. [94] compared several new and existing mixed-integer programming models with special attention paid to multivariate non-separable functions and the traditional SOS2 formulation of continuous piecewise-linear functions, which does not include binary variables. The comparison is made based on their theoretical properties and their relative computational performance.

Some other related work is detailed in the following paragraph. Sherali and Tuncbilek [37] proposed a generic branch-and-bound algorithm for globally optimizing continuous polynomial programming problems, which employs constructed linear bounding problems using a reformulation linearization technique (RLT) in concert with a suitable partitioning strategy that guarantees the convergence of the over-

all algorithm. Sherali and Wang [95] presented a global optimization approach for solving non-convex factorable programming problems. A branch-and-bound procedure with a suitable partitioning scheme and two levels of relaxations is proposed, ensuring convergence to a global optimum. At the first level, the lower bounding relaxation involves a tight non-convex polynomial approximation; an LP relaxation is then constructed for the resulting polynomial program via a RLT procedure. Sherali and Ganesan [96] presented two pseudo-global optimization approaches for solving formidable constrained optimization problems such as the containership design model. These approaches are based on iteratively using the response surface methodology or curve-fitting procedures, to construct polynomial programming approximations; along with certain global optimization schemes such as the RLT for effectively solving polynomial programming problems. Zhang and Wang [97] approximated a nonlinear objective function subject to linear constraints by a continuous piecewise linear function expressed as a lattice representation. A systematic approach for this class of problems is proposed that determines an approximately globally optimal solution. Chrysanthos et al. [98] showed the solution to the problem of optimizing the distribution of a limited supply of lift gas, using four mathematical formulations for maximizing over piecewise linear functions. A comparative study demonstrates that each of the four models is sufficient to solve the problem to global optimality. The computational performance of each model is also investigated.

2.4 Crashworthiness and Surrogate-Based Global Optimization Methods in Crashworthiness

When a product like an automobile is developed, the safety system design becomes one of the major attributes. Crashworthiness [1] is the ability of a structure to protect its occupants during an impact in such a way that the structure of a car must be able to attenuate the crash force when impact occurs.

There exist multiple crash scenarios that need to be analyzed during an automotive crashworthiness study: full front impact, 50% front offset impact, roof crush model, and side impact.

However this is considered computationally complex due to the significant number of simulations required before an optimal design can be attained. Therefore the study of different approximation or surrogate models methods have been intensively examined, especially in vehicle crashworthiness for occupant safety design. In 2001, Gu et al. [99] presented a non-linear response surface-based safety optimization process applied to the vehicle crash safety design of side impact. In 2005, Yang et al. [100] studied five response surface methods using a real-world frontal impact design problem as an example. And more recently in 2008, Liao et al. [101] proposed a multi-objective optimization procedure for the multi-objective design of vehicles crashworthiness using simple stepwise regression models. Due to the importance and continuous demand of vehicle safety designs, the proposed method is demonstrated on designing an automotive safety system for a major US automaker in the following chapter.

[1] *http://en.wikipedia.org/wiki/Crashworthiness*

2.5 Contribution

After studying and reviewing the related literature to the proposed method, this section emphasizes the contribution of this research.

Some important aspects summarized from the previous sections are as follows:

- In comparison with previous models, the employment of MARS is considered relatively new.
- Although MARS has been used as a surrogate model in different optimization approaches, literature reports its applications only on well known unconstrained optimization test functions and low dimensional examples.
- MARS approximations have been used in numerical solutions to large-scale optimization problems [6, 7]. However these cases were assumed to be convex.
- As mentioned before, the main drawback or inconvenience of other GO standard procedures like evolutionary algorithms and population-based strategies such as the extensively applied genetic algorithm, is that they require a significant number of simulations of the objective and constraint functions, which is frequently impractical in terms of computational times.
- In some of the previously presented studies, researchers have developed global optimization approaches for functions that are treated as expensive black-box problems, i.e. there is not prior knowledge or property of the function.
- Existing mixed integer programming models of multivariate non-separable functions do not include binary explanatory variables.

The contribution of this research is a novel deterministic global optimization method based on mixed integer linear programming to solve non-convex piecewise linear functions generated by a modified version of MARS subject to constraints that include both linear regression models and piecewise linear MARS models. Due to the

linearization of the interaction terms in the modified version of MARS, the proposed method can easily handle possible non-linearities of objective and constraint functions.

A new mixed integer linear programming problem that can optimize the piecewise linear MARS model is formulated by introducing binary integer variables and a new set of inequalities, and is adjusted to enable flexibility to handle both continuous and categorical variables.

The use of piecewise linear MARS as a surrogate method makes the global optimization easier to solve while reducing computationally expensive simulations; consequently, the proposed method has potential for optimizing other complex systems.

The proposed method is compared with a genetic algorithm, which is a prevalent optimization technique extensively used. The method in this dissertation performs substantially better and where the CPU time is negligible.

A validation procedure based on six evaluators shows the success of the proposed method identifying Pareto optimal solutions.

An aspect that differentiates the developed method with others surrogate based optimization methods is the assumption of having static data; that is, there is no certainty that additional data can be gathered, and this is the case in real-world problems such as crash simulations or patients under certain treatments, in which experimentation becomes unavailable.

A rounding approach to incorporate discrete variables in a customized genetic algorithm is developed.

CHAPTER 3

GLOBAL OPTIMIZATION OF A NONCONVEX PIECEWISE LINEAR REGRESSION SPLINES

This research presents a method for globally optimizing computationally expensive computer simulations through solving non-convex piecewise linear functions generated by a modified version of multivariate adaptive regression splines (MARS). The advantage of using MARS as a surrogate model for this method is that MARS is intended for high dimensional data that can include interaction and curvature, consequently more flexible than a linear regression model. Also, it is particularly useful due to the contribution made by Shih in 2006 [9] and later improved by Martinez in 2011 [102] of transforming the interaction terms to piecewise linear approximations, enabling a much easier and faster search of a global optima. A mixed integer linear programming model that can optimize the piecewise linear MARS function is presented, and the algorithm used for solving this mathematical programming model is branch-and-bound.

This section presents the original MARS statistical modeling method followed by the modified version of the original MARS function. Then, the mixed integer linear programming formulation of the piecewise linear version of MARS is presented. This global optimization approach is shown in several instances, with a single non-convex piecewise linear MARS function and with a non-convex piecewise linear MARS function subject to both piecewise linear MARS models and linear regression models, where the MILP model is adjusted in order to enable flexibility for handling continuous and categorical variables.

3.1 Background

3.1.1 Original Multivariate Adaptive Regression Splines (MARS)

Multivariate adaptive regression splines of Friedman is a forward-backward stepwise subset selection procedure that builds a model using a set of spline basis functions that best fit the data.

The MARS model terms are based on truncated linear functions, where the univariate terms are piecewise linear, and the interaction terms, which are generated by taking products of univariate indicator factors, are not, generating this non-linearities on the function.

The MARS approximation has the form:

$$\hat{f}_M(x, \beta) = \beta_0 + \sum_{m=1}^{M} \beta_m B_m(x), \tag{3.1}$$

where x is an n-dimensional vector of explanatory variables, β_0 is the intercept coefficient, which is the mean of the response values, M is the number of linearly independent basis functions, β_m is the unknown coefficient for the mth basis function, and $B_m(x)$ is a basis function that utilizes truncated linear functions.

The *univariate basis functions* are truncated linear functions of the form:

$$b^+(x; k) = [+(x - k)]_+ \qquad \text{or} \qquad b^-(x; k) = [-(x - k)]_+, \tag{3.2}$$

where $[q]_+ = \max\{0, q\}$, x is a single explanatory variable, and k is the corresponding univariate knot, where the approximation bends to model curvature.

The *interaction basis functions* are formed as a product of two or more truncated linear functions; hence the interaction of two or more variables is implied.

The basis function is of the following form:

$$B_m(x) = \prod_{l=1}^{L_m} [s_{l,m}(x_{v(l,m)} - k_{l,m}]_+, \tag{3.3}$$

where L_m indicates the order of the interaction terms in the mth basis function, which is defined by the user, and is typically set up to three-way interactions, $x_{v(l,m)}$ is the explanatory variable corresponding to the lth truncated linear function in the mth basis function, and $k_{l,m}$ is the knot value corresponding to $x_{v(l,m)}$. The value $s_{l,m}$ is the direction that the truncated linear basis function can take, either +1 or -1.

Figure 3.1 from Shih [103] represents two-way interaction terms for Original MARS and for Piecewise Linear version of MARS, which is explained in the next section.

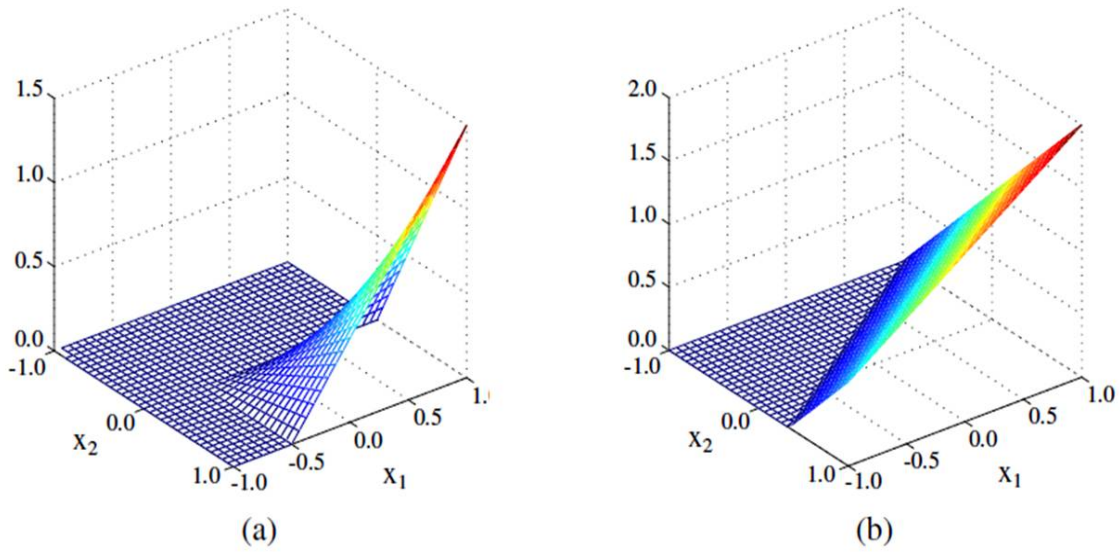

Figure 3.1. Two-way interaction terms for Original MARS (a) and for Piecewise Linear version of MARS (b).

3.1.2 Piecewise Linear version of MARS

In 2006, Shih [9] developed the convex version of MARS, where he included a transformation of the interaction terms to a linear combination of variables; later Martinez [102] utilized the proposed transformation of Shih to create a non-convex piecewise linear version of MARS.

The non-linearities generated by the multiple variables of the MARS interaction terms can be modified using the previously mentioned transformation method. In order to do this, the form of the basis function $B_m(x)$ in (3.3) needs to be modified to a new-one dimensional variable with basis function forms that can be treated like univariate basis functions.

Each interaction basis function can be modified using the following transformation term z_m:

$$z_m = a_{0,m} + \sum_{l=1}^{L_m} a_{l,m} x_{v(l,m)}, \tag{3.4}$$

where

$$a_{0,m} = \sum_{l=1}^{L_m} \frac{s_{l,m} k_{l,m}}{s_{l,m} k_{l,m} - 1} \qquad a_{l,m} = \frac{s_{l,m}}{1 - s_{l,m} k_{l,m}} \tag{3.5}$$

The terminology previously defined is the same for these two equations.

Using the transformation in equations (3.4) and (3.5), the general form of a piecewise linear interaction basis function is then defined as:

$$B_m(x) = [\phi_m z_m]_+, \tag{3.6}$$

where

$$\phi_m = \begin{cases} +1 (or +) \\ -1 (or -) \end{cases} \tag{3.7}$$

The parameter ϕ_m represents the direction of the linear combination of variables.

3.2 Global Optimization of a Non-Convex Piecewise Linear Regression Splines Function

3.2.1 Single Non-Convex Piecewise Linear MARS Function

Given the linearization of the MARS interaction terms resultant of the transformation presented in the previous section, the general formulation of the non-convex piecewise linear MARS optimization problem can be defined as follows:

$$\text{Maximize } f(x) = \beta_0 + \sum_{m=1}^{M} \beta_m B_m(x), \tag{3.8}$$

$$\text{Subject to } 0 \leq B_m(x) \leq u_m, \qquad \forall m = 1, ..., M \tag{3.9}$$

$$l_j \leq x_j \leq u_j, \qquad \forall j = 1, ..., n \tag{3.10}$$

where the general problem (3.8) is to be maximized (or minimized depending on the form of the problem) the sum of piecewise linear basis functions $B_m(x)$, subject to linear constraints bounded to an upper constant u_m and a non-negativity restriction for all the basis functions (3.9). β_0 is the intercept coefficient, β_m is the coefficient of the piecewise linear basis functions $B_m(x)$, and x represents the vector of the explanatory variables bounded by a lower and upper constant value, l_j and u_j respectively.

The piecewise linear MARS optimization problem can be formulated by adding binary integer variables and a new set of inequalities to the model, yielding a Mixed Integer Linear Programming (MILP) problem.

Let: y_m^+ and y_m^- represent the binary integer variables, which are defined by the following conditions:

$$y_m^+ = \begin{cases} 1, & \text{if } z_m \geq 0 \\ 0, & \text{otherwise,} \end{cases} \qquad y_m^- = \begin{cases} 1, & \text{if } z_m \leq 0 \\ 0, & \text{otherwise,} \end{cases} \tag{3.11}$$

$l_m = -u_m$, indicate a constant lower bound of the z_m term, and

B_m represent the function value of the $B_m(x)$.

The MILP problem is expressed with the following equations and inequalities:

$$\text{Maximize } f(x) = \beta_0 + \sum_{m=1}^{M} \beta_m B_m, \tag{3.12}$$

$$\text{Subject to } z_m = \begin{cases} a_{0,m} + \sum_{l=1}^{L_m} a_{l,m} x_{v(l,m)}, & \text{if interaction term} \\ [s_m(x_m - k_m)]_+, & \text{if univariate term} \end{cases} \qquad \forall m = 1, ..., M, \tag{3.13}$$

$$z_m \leq y_m^+ u_m \qquad \forall m = 1, ..., M, \tag{3.14}$$

$$z_m \geq y_m^- l_m \qquad \forall m = 1, ..., M, \tag{3.15}$$

$$y_m^+ + y_m^- = 1 \qquad \forall m = 1, ..., M, \tag{3.16}$$

Observe that the term z_m represents the transformed linear combination of variables for the interaction terms and the truncated linear function for the univariate terms.

The following standard bounds and integrality constraints (3.17)-(3.21) are added to the model.

$$l_j \leq x_j \leq u_j, \qquad \forall j = 1, ..., n \tag{3.17}$$

$$x \in \Re^n, \tag{3.18}$$

$$y_m^+, y_m^- \in \{0, 1\} \qquad \forall m = 1, ..., M, \tag{3.19}$$

$$z_m \text{ free} \qquad \forall m = 1, ..., M, \tag{3.20}$$

$$B_m \geq 0 \qquad \forall m = 1, ..., M. \tag{3.21}$$

In conjunction with the previous formulation, the following subset of constraints based upon the parameter ϕ_m, which as was mentioned before, represents the direction of the transformed linear combination of variables, is also added to the model.

$$0 \leq B_m \leq u_m y_m^{\phi_m} \quad \forall m = 1, ..., M, \tag{3.22}$$

$$\phi_m z_m \leq B_m \leq \phi_m z_m + u_m y_m^{-\phi_m} \quad \forall m = 1, ..., M. \tag{3.23}$$

It is assumed that the parameter ϕ_m is $+1$ for the univariate terms.
By (3.22)-(3.23), B_m is equivalent to:

$$B_m = \begin{cases} \phi_m z_m, & \text{if } \phi_m z_m \geq 0 \\ 0, & \text{otherwise,} \end{cases} \tag{3.24}$$

Equations (3.13) and (3.16), and inequalities (3.14)-(3.15), (3.22)-(3.23), add up to six specific constraints for each basis function B_m. From now on, consider the term C^m as the set of feasible basis functions of a piecewise linear MARS model,

$$C^m = \{(x, B_m) | \exists (z_m, y_m^+, y_m^-) : s.t. (x, B_m, z_m, y_m^+, y_m^-) satisfy (3.13) - (3.23)\} \tag{3.25}$$

3.2.2 Non-Convex Piecewise Linear MARS Function Subject to Piecewise Linear MARS Models and Linear Regression Models

The previous section showed the method for optimizing a single piecewise linear MARS function. However, in order to enable a wider use of the proposed method, the MILP formulation is adjusted in such a way that not just the objective function but also the constraints of a complex system can be modeled using piecewise linear MARS functions, therefore allowing an easier and faster optimization.

Although piecewise linear MARS functions are of main interest in this study, the use of other functions such as linear regression models can also be implemented in a simple manner.

Therefore, the response surface models used for the objective and the constraint functions of the proposed formulation, are built using piecewise linear MARS models and linear regression models, utilizing the coefficient of determination R^2 and the analysis of residual plots as performance indicators to select and describe how well a regression line fits a set of data.

The MILP problem can now be expressed as follows:

$$\text{Maximize } \hat{g}_1(x) \tag{3.26}$$

$$\text{Subject to } \hat{g}_i(x) \leq b_i, \qquad \forall i = 2, ..., G, \tag{3.27}$$

$$l_j \leq x_j \leq u_j, \qquad \forall j = 1, ..., n. \tag{3.28}$$

where G represent the number of constraint functions plus one and n the number of explanatory variables.

The general problem, (3.26), is to maximize (or minimize, depending on the form of the problem) a function approximated either by a linear regression model (3.29) or by a piecewise linear MARS model (3.30), subject to linear constraints that can also be approximated by a linear regression model or a piecewise linear MARS model. These restrictions are written as (3.27), where b represents a vector of upper constant bounds, x represents the vector of explanatory variables, which are bounded by lower and upper constant values.

The linear models $\hat{g}_i(x)$ can be approximated by a linear regression model, which has the following form:

$$\text{Linear Regression Function } \hat{g}_i(x) = \beta_{i0} + \sum_{j=1}^{ni} \beta_{ij} x_j, \tag{3.29}$$

or they can be approximated by the form of a piecewise linear MARS function:

$$\text{Piecewise Linear MARS Function } \hat{g}_i(x) = \beta_{i0} + \sum_{m=1}^{Mi} \beta_{im} B_m \tag{3.30}$$

$$\text{Subject to } (x, B_m) \in C^m \qquad \forall m = 1, ..., Mi. \tag{3.31}$$

Observe that every linear model approximated by (3.30) is subject to the set of constraints (3.31). The previous restrictions (3.13)-(3.16), (3.22)-(3.23), and the general bounds (3.18)-(3.21) complete the MILP formulation of a non-convex piecewise linear MARS function subject to linear constraint models.

CHAPTER 4

APPLICATIONS AND COMPUTATIONAL RESULTS

The present chapter demonstrates the proposed method employed in two different case studies. The first case is a four-dimensional inventory test problem in which a single non-convex piecewise linear MARS function is optimized, obtaining a global optimization solution. The second case is an automotive crash safety system design example for a major US automaker, in which a non-convex piecewise linear MARS function subject to multiple linear constraint functions is optimized. The statistical analysis for both cases has been provided by Martinez [104].

This research focuses and emphasizes its interest on this last case study, thus it also shows a sensitivity analysis of its corresponding mixed integer linear programming model. The objective function of the MILP model is approximated by a non-convex piecewise linear MARS function and also by a stepwise linear regression function, both cases are globally optimized.

4.1 Non-Convex Minimization Inventory Test Problem

An inventory forecasting Stochastic Dynamic Programming (SDP) problem with four state variables is used to illustrate the proposed global optimization method. The objective of this SDP problem studied by Chen in 1999 [105] is to minimize expected inventory cost, involving backorder and inventory holding costs, which uses state variables for the inventory level of two items and the forecast for future customer demands for each item.

A non-convex piecewise linear MARS function is approximated by modifying the coefficients of the convex version of MARS, so that they clearly violated convexity. The non-convex piecewise linear MARS function is built setting the interaction terms up to three-way interactions and using 125 training data points for the last time period of the inventory forecasting SDP problem. The resultant approximation consists of a set of 11 basis functions.

The corresponding mixed integer linear programming problem is obtained by using the formulation presented in section 3.2; the objective function is set as to minimize the sum of the piecewise linear MARS functions multiplied by their respective coefficients, which in this case represent an optimal minimum cost of the inventory system, subject to six different constraints for each of the basis functions and restricted by the corresponding general bounds.

The MILP formulation for the minimization inventory test problem is shown as follows:

Minimize: $-176.5317 + 45.3915 * B_1 - 68.0405 * B_2 + 8.6999 * B_3 - 85.2167 * B_4 - 187.9091 * B_5 + 98.5907 * B_6 - 6.0805 * B_7 - 124.1187 * B_8 - 74.1405 * B_9 - 124.3142 * B_{10} - 144.6767 * B_{11}$

Subject to:

$$\text{Set of constraints for } B_1 \begin{cases} c1 : 0.6667x_2 + z_1 = 0.3333 \\ c2 : z_1 - 999y_1^+ \leq 0 \\ c3 : z_1 + 999y_1^- \geq 0 \\ c4 : y_1^+ + y_1^- = 1 \\ c5 : 0 \leq B_1 \leq +999y_1^+ \\ c6 : z_1 \leq B_1 \leq z_1 + 999y_1^- \end{cases}$$

$$\text{Set of constraints for } B_{11} \begin{cases} c61 : x_2 + z_{11} = 0 \\ c62 : z_{11} - 999y_{11}^+ \leq 0 \\ c63 : z_{11} + 999y_{11}^- \geq 0 \\ c64 : y_{11}^+ + y_{11}^- = 1 \\ c65 : 0 \leq B_{11} \leq +999y_{11}^+ \\ c66 : z_{11} \leq B_{11} \leq z_{11} + 999y_{11}^- \end{cases}$$

$$-1 \leq x_j \leq 1 \qquad \forall j = 1, ..., 4,$$

$$y_m^-, y_m^+ \in \{0, 1\} \qquad \forall m = 1, ..., 11,$$

$$z_m \text{ free} \qquad \forall m = 1, ..., 11,$$

$$B_m \geq 0 \qquad \forall m = 1, ..., 11.$$

For simplicity purposes, the constraints for only the 1st and the 11th basis functions are presented.

The formulation of the MILP problem is generated by a C-programming code that is executed on a Dual 2.6 GHz Athlon workstation. Finally, the global optimiza-

tion solution to the inventory test problem is obtained by employing branch-and-bound using CPLEX as a solver. The computational time to achieve the solution of this problem is just a few seconds.

The non-convex piecewise linear MARS function objective value is equal to -974.96, which represents the minimum holding and backorder inventory costs. The rest of the CPLEX results are shown in Tables 4.1 and 4.2:

Table 4.1. Solution for the explanatory variables of Inventory Test Problem

x_j
$x_1 = -1$
$x_2 = -1$
$x_3 = +1$
$x_4 = +1$

The optimal solution presented is scaled, but it would represent how much to order of each item in order to minimize the cost for operating the system.

Table 4.2. Solutions for the terms z_m and the basis functions B_m

z_m	ϕ_m	B_m
$z_1 = +1.000$	$+1$	$B_1 = 1.000$
$z_2 = -3.000$	$+1$	$B_2 = 0.000$
$z_3 = +2.000$	-1	$B_3 = 0.000$
$z_4 = +0.667$	-1	$B_4 = 0.000$
$z_5 = -3.333$	-1	$B_5 = 3.333$
$z_6 = -2.000$	-1	$B_6 = 2.000$
$z_7 = -1.000$	$+1$	$B_7 = 0.000$
$z_8 = +3.000$	$+1$	$B_8 = 3.000$
$z_9 = +1.000$	$+1$	$B_9 = 1.000$
$z_{10} = -2.000$	$+1$	$B_{10} = 0.000$
$z_{11} = +1.000$	$+1$	$B_{11} = 1.000$

All constraints are satisfied, therefore the above optimal solution holds for the whole domain of the function, meaning that it is a globally optimal.

4.2 Automotive Crash Safety System Design Case Study

Stepwise regression (SR) methods have been commonly used as meta-models to approximate computationally expensive complex systems such as safety related functions in automotive crash analysis, multi-objective optimization for crash safety design of vehicles, frontal impact design problems, and crash safety design of vehicles see Yang et al. [106], Gu et al. [99], Yang [100], and Liao et al. [101], respectively.

Stepwise regression helps finding the best subset of prediction variables by adding or eliminating variables that provide a better fit or do not improve the model, respectively. It consists of two procedures, a forward stepwise regression, which develops a sequence of simple linear regression models, adds variables based on a certain criterion and ends identifying the best single linear regression model. A backward stepwise regression procedure is alternatively executed, which eliminates variables based on statistical criteria and terminates when there is no other variable to be dropped.

Consequently in the following case study, the proposed method is applied to an automotive crash safety system design example using stepwise linear regression (SLR) and piecewise linear MARS as meta-models to approximate the functions, where the objective is to minimize the function subject to constraints and bounds of design variables.

4.2.1 Overview of the Problem and Formulation

The automotive crash safety system design case study consists of 33 input variables. Examples of these variables are passenger airbag, retractor torsion bars, etc.

where 23 of them are continuous, 7 are 2-level categorical variables (highlighted in gray), and 3 are 4-level categorical variables (highlighted in yellow). It includes 51 output variables that represent the objective function, which is to be minimized, and 50 constraints. Tables 4.3 and 4.4 provide lower and upper bounds of the input (explanatory) variables, and the right hand side (RHS) values of the inequality constraints, respectively.

The case study is also conformed by a set of data that contains 200 points and an additional set that includes 1249 points (Data set 2). The former set of data (Data set 1) is used to build the linear models.

The piecewise linear MARS models and the stepwise linear regression models are built utilizing scaled values for the variables, based on the mid-range and the half-range of the set of data variable values (refer to Appendix A). Therefore the optimization problem provides a scaled solution, which is later converted to the original units of the data (this solution is called un-scaled). With this nominal scale, the values of the variables lie between -1 and 1.

As was previously mentioned, the problem consists of 10 categorical variables, 7 of them are directly treated as binary variables (also known as dummy variables), since they can only take on two different possible values. For the other three categorical variables, a dummy coding is required. Variables number 1 (PAB Shape), 22 (Passenger airbag lower tether location), and 33 (Passenger airbag upper tether location) can only take on one of the following possible values 1, 2, 3 or 4; each level is represented with a -1,1 binary variable, i.e., only one of the binary variables can take on the value of positive 1. Table 4.5 illustrates an example of how these variables are coded.

Example in Table 4.5, indicates that the value for variable 1 in this case is 3, since is the one that has the positive value. This dummy coding is simplified by considering

Table 4.3. Information on explanatory variables

No.	Description	Lower Bound	Upper Bound	Variable Type
1	PAB Shape	1	4	4-level cat.
2	PAB Size	-0.2	1.0	continuous
3	Buckle pretensioner flag	0	1	2-level cat.
4	Retractor pretensioner flag	0	1	2-level cat.
5	Adaptive belt load limiter flag	0	1	2-level cat.
6	Crash locking tongue flag	0	1	2-level cat.
7	Knee airbag flag	0	1	2-level cat.
8	Passenger airbag adapt vent flag	0	1	2-level cat.
9	Heel stopper flag	0	0	2-level cat.
10	Buckle pretensioner pull in (m)	0.06	0.1	continuous
11	Buckle pretensioner time to fire (s)	0.008	0.013	continuous
12	Retractor pretensioner pull in (m)	0.06	0.1	continuous
13	Retractor pretensioner time to fire (s)	0.008	0.013	continuous
14	Retractor torsion bar force level-1	2000	3000	continuous
15	Retractor torsion bar force level-2	2000	3200	continuous
16	Retractor torsion bar dispalcement interval-05	0.05	0.3	continuous
17	Retractor torsion bar dispalcement interval-50	0.05	0.3	continuous
18	Knee airbag time to fire (s)	0.013	0.2	continuous
19	Knee airbag inflator power	0.75	1.5	continuous
20	Knee airbag vent size (mm)	0	15	continuous
21	Passenger airbag lower tether length (mm)	0.4	0.52	continuous
22	Passenger airbag lower tether location	1	4	4-level cat.
23	Passenger airbag time to fire (s)-u05	0.01	0.013	continuous
24	Passenger airbag Z-Scale	0.8	1.2	continuous
25	Passenger airbag adaptive vent size (mm)	40	120	continuous
26	Passenger airbag time to fire (s)-b05	0.01	0.1	continuous
27	Passenger airbag time to fire (s)-b50	0.01	0.1	continuous
28	Passenger airbag time to fire (s)-u05	0.02	0.1	continuous
29	Passenger airbag time to fire (s)-u50	0.02	0.1	continuous
30	Passenger airbag fixed vent size (mm)	40	80	continuous
31	Passenger airbag inflator power	0.8	1.2	continuous
32	Passenger airbag upper tether length (mm)	0.4	0.52	continuous
33	Passenger airbag upper tether location	1	4	4-level cat.

Table 4.4. Information on the objective function and constraints

No.	Name	Objective /RHS	No.	Name	RHS
	Obj-pb05-RRS	Minimize			
1	constr-far50-ChestD	≤ 1	26	constr-pb50-HIC	≤ 1
2	constr-far50-ChestG	≤ 1	27	constr-pb50-Head-IP-min	≥ 1
3	constr-far50-Chest-IP-min	≥ 1	28	constr-pb50-NeckFzMax	≤ 1
4	constr-far50-FemurL	≤ 1	29	constr-pb50-NeckFzMin	≤ 1
5	constr-far50-FemurR	≤ 1	30	constr-pb50-Nij	≤ 1
6	constr-far50-HIC	≤ 1	31	constr-pu05-ChestD	≤ 1
7	constr-far50-Head-IP-min	≥ 1	32	constr-pu05-ChestG	≤ 1
8	constr-far50-NeckFzMax	≤ 1	33	constr-pu05-Chest-IP-min	≥ 1
9	constr-far50-NeckFzMin	≤ 1	34	constr-pu05-FemurL	≤ 1
10	constr-far50-Nij	≤ 1	35	constr-pu05-FemurR	≤ 1
11	constr-pb05-ChestD	≤ 1	36	constr-pu05-HIC	≤ 1
12	constr-pb05-ChestG	≤ 1	37	constr-pu05-Head-IP-min	≥ 1
13	constr-pb05-Chest-IP-min	≥ 1	38	constr-pu05-NeckFzMax	≤ 1
14	constr-pb05-FemurL	≤ 1	39	constr-pu05-NeckFzMin	≤ 1
15	constr-pb05-FemurR	≤ 1	40	constr-pu05-Nij	≤ 1
16	constr-pb05-HIC	≤ 1	41	constr-pu50-ChestD	≤ 1
17	constr-pb05-Head-IP-min	≥ 1	42	constr-pu50-ChestG	≤ 1
18	constr-pb05-NeckFzMax	≤ 1	43	constr-pu50-Chest-IP-min	≥ 1
19	constr-pb05-NeckFzMin	≤ 1	44	constr-pu50-FemurL	≤ 1
20	constr-pb05-Nij	≤ 1	45	constr-pu50-FemurR	≤ 1
21	constr-pb50-ChestD	≤ 1	46	constr-pu50-HIC	≤ 1
22	constr-pb50-ChestG	≤ 1	47	constr-pu50-Head-IP-min	≥ 1
23	constr-pb50-Chest-IP-min	≥ 1	48	constr-pu50-NeckFzMax	≤ 1
24	constr-pb50-FemurL	≤ 1	49	constr-pu50-NeckFzMin	≤ 1
25	constr-pb50-FemurR	≤ 1	50	constr-pu50-Nij	≤ 1

Table 4.5. Example of dummy coding

Variable 1				
Possible values:	1	2	3	4
Binary values:	-1	-1	1	-1

the number of levels minus one, i.e. if all the present values have a -1, the absent

variable is the one chosen, having to use the number of levels minus one binary variables only.

The same encoding is used for the categorical variables 22 and 33. However, even though these variables can also take on four possible values based on initial variable information, the set of 200 points is missing a level for each of these variables, therefore only two binary variables for each of them are needed. With the incorporation of these binary variables, the number of variables increases from 33 to 37, where 14 of them are -1,1 binary variables (three required to represent variable 1, two for variable 22, two for variable 33, and seven 2-level binary variables).

The objective function was first approximated using a stepwise linear regression model, resulting in a coefficient of determination equal to 0.77, and the residual plot presented in Figure 4.1.

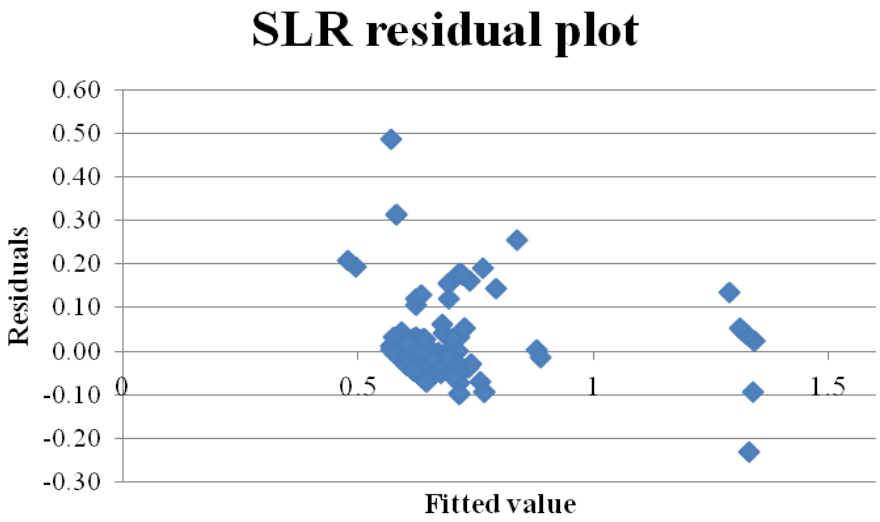

Figure 4.1. SLR Residual Plot.

Although the R^2 value is relatively high, and the residual plot does not show a clear evidence of curvature, the objective was also approximated using a piecewise linear MARS (PL-MARS) model with a coefficient of determination equals to 0.90, indicating that the data points fit a curve better than a line. Its corresponding residual plot is shown in Figure 4.2.

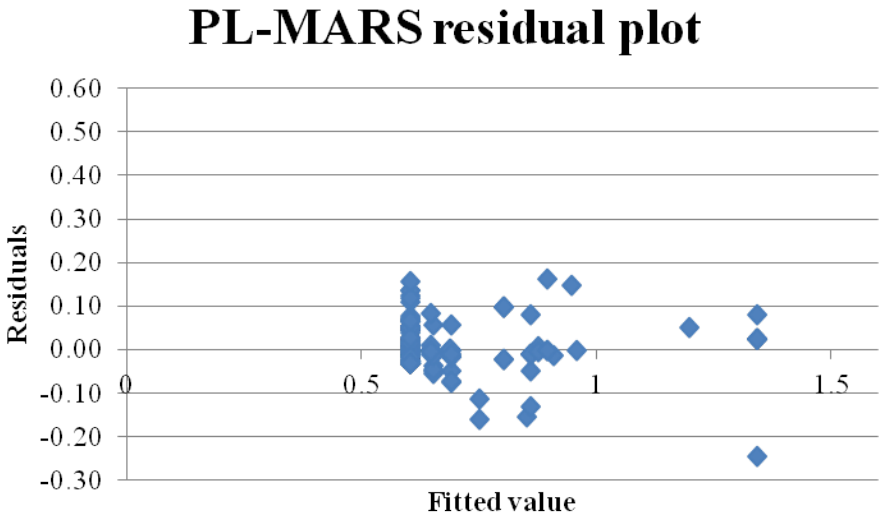

Figure 4.2. PL-MARS Residual Plot.

Since there is no any other information provided about these data, and that the underlying function is unknown, the solution for the optimization problem is later presented with the usage of both models.

Stepwise linear regression models are constructed in order to study the behavior and conditions of the rest of the output variables that is, 50 constraints (these terms are used interchangeably). Based on the coefficient of determination R^2 and the analysis of residual plots, 10 out of 50 models show curvature and low R^2 values ($<$ 0.70), indicating that a regression line does not fit the data well for those 10 models,

therefore piecewise linear MARS approximations are necessary. Table 4.6 provides the information on the output variables regarding the R^2 obtained based on the model used to approximate these constraints. The highlighted values in Table 4.6 indicate the R^2 values that are less than < 0.70 when approximating these constraints by a SLR function, however their corresponding residual plots did not show curvature, so PL-MARS models were not built. The piecewise linear MARS functions are restricted to up to two-way interaction terms. The number of basis functions for each of the 10 piecewise linear MARS approximations varies from 6 to 10.

Table 4.6. R^2 for output variable models

No.	Model	R-squared	No.	Model	R-squared
1	SLR	0.5681	26	SLR	0.7203
2	SLR	0.3723	27	SLR	0.8483
3	PL-MARS	0.7812	28	SLR	0.5822
4	PL-MARS	0.9176	29	SLR	0.6535
5	PL-MARS	0.9317	30	SLR	0.6287
6	SLR	0.4728	31	SLR	0.4373
7	SLR	0.7831	32	SLR	0.8376
8	PL-MARS	0.7291	33	SLR	0.8795
9	SLR	0.5542	34	SLR	0.5538
10	PL-MARS	0.7063	35	SLR	0.5607
11	SLR	0.9438	36	SLR	0.8060
12	SLR	0.8488	37	SLR	0.8965
13	SLR	0.9739	38	SLR	0.6175
14	PL-MARS	0.9826	39	SLR	0.8003
15	SLR	0.8661	40	SLR	0.6192
16	PL-MARS	0.7976	41	PL-MARS	0.8185
17	SLR	0.8196	42	SLR	0.6566
18	SLR	0.8937	43	PL-MARS	0.9389
19	SLR	0.8403	44	SLR	0.9337
20	SLR	0.8348	45	SLR	0.9358
21	PL-MARS	0.7767	46	SLR	0.6129
22	SLR	0.5143	47	SLR	0.9051
23	SLR	0.7088	48	SLR	0.6023
24	SLR	0.8582	49	SLR	0.7027
25	SLR	0.8418	50	SLR	0.6842

Based on the formulation criteria proposed and explained in Chapter 3, the MILP model for this case study results as follows:

$$\text{Minimize } \hat{g}_1(x) \tag{4.1}$$

$$\text{Subject to } \hat{g}_i(x) \leq b_i, \qquad \forall i = 2, ..., 51, \tag{4.2}$$

$$-1 \leq x_l \leq 1, \qquad \forall l = 1, ..., 37, \tag{4.3}$$

$$x \in \Re^{37}. \tag{4.4}$$

where b represents the vector of upper constant bounds of the 50 constraints, based on the information given in Table 4.4. Note that the total number of variables, which includes the variables representing the levels of the categorical variables, is now in set L.

Forty of the linear models $\hat{g}_i(x)$ are approximated by a stepwise linear regression model:

$$\text{Linear Regression Function } \hat{g}_i(x) = \beta_{i0} + \sum_{l=1}^{Li} \beta_{il} x_l, \tag{4.5}$$

while the other ten constraints are approximated by the form of a piecewise linear MARS function:

$$\text{Piecewise Linear MARS Function } \hat{g}_i(x) = \beta_{i0} + \sum_{m=1}^{Mi} \beta_{im} B_m \tag{4.6}$$

$$\text{Subject to } (x, B_m) \in C^m \qquad \forall m = 1, ..., Mi. \tag{4.7}$$

This MILP problem is solved in two instances, when the objective function is approximated using a stepwise linear regression (SLR) model and when it is approximated by a piecewise linear MARS model subject to its corresponding set of feasible basis functions. Each objective function model is now presented:

Objective function approximated by a SLR model

Minimize: $1.18824 - 0.00523 * x_{1c} - 0.02652 * x_4 + 0.04668 * x_9 - 0.02358 * x_{10} + 0.01403 * x_{11} + 0.13628 * x_{14} - 0.04167 * x_{16} + 0.03350 * x_{17} + 0.00716 * x_{20} - 0.01904 * x_{21} + 0.04536 * x_{23} - 0.09035 * x_{24} - 0.02362 * x_{25} - 0.10452 * x_{30} - 0.35430 * x_{32} - 0.00842 * x_{33b}$

Objective function approximated by a Piecewise Linear MARS model

Minimize: $0.778713 + 0.183098 * B_1 + 0.508493 * B_2 + 0.082256 * B_3 + 0.098260 * B_4 + 0.290244 * B_5 - 0.087319 * B_6 + 0.184992 * B_7 - 0.043532 * B_8$

Refer to Appendix B for the complete function. Appendix C displays the graphs of the objective function using both of the previous models.

Due to the scalarization used for the variables and the presence of categorical variables that the problem contains, an additional set of binary variables with possible values of zero and one, a set of equality constraints and a set of inequality constraints, are required.

Let the explanatory variables j=1, 3-9, 22 and 33 be in set K.

$L(j)$=k's that have different levels of variable j, e.g. variable j=1, $k(1)$ = 1,2,3.

Therefore, in order to ensure that the variables that require to be binary variables can take on only the values of -1 or +1, the following equality is needed:

$$x_l - 2w_l = -1, \qquad \forall l \in L(j), j \in K \tag{4.8}$$

$$w_l \in \underline{\mathrm{B}} \qquad \forall l \in L(j), j \in K \tag{4.9}$$

where w_l represents the 0-1 binary variable associated with the lth variable.

Finally, the only restrictions missing in order to complete the formulation, are the well known packing-type constraints. These constraints ensure that only one of

the possible values is chosen, and are required for all of the categorical variables with more than two levels, i.e., variables 1, 22 and 33 in this case.

The general form of this constraint is given by (4.10):

$$\sum_{l \in L(j)} w_l \leq 1, \qquad \forall j \in K. \tag{4.10}$$

The three packing constraints needed for this case study are also added to the previous MILP model and are expressed as follows:

$$w_{1a} + w_{1b} + w_{1c} \leq 1, \tag{4.11}$$

$$w_{22a} + w_{22c} \leq 1, \tag{4.12}$$

$$w_{33b} + x_{33c} \leq 1. \tag{4.13}$$

Note that the absent variable for all the cases (j=1, 22 and 33) is the variable representing the value of 4, while the missing level for variables 22 and 33 is the variable representing the value of 2 and 1 respectively.

The MILP model is as well generated by a C-programming code, which is executed on a Dual 2.6 GHz Athlon workstation. The MILP problem is solved by a branch-and-bound algorithm using CPLEX as a solver.

The computational results are presented in Table 4.7 where the CPLEX solution value of the explanatory variables (scaled and un-scaled) is shown, using both the SLR model and the PL-MARS model for the objective function.

Tables 4.8 and 4.9 report a globally optimal objective function value, and the value of the output variables (constraints) for both instances, where it can be noted that all the constraints are satisfied. These tables also indicate what model was used to approximate each of the constraints ('Model used' column).

Table 4.7. Scaled and un-scaled solution obtained using SLR model and PL-MARS model

SLR model solution			
ID	**Scaled**	**ID**	**Un-scaled**
1a	-1.000000	1	3.00000
1b	-1.000000		
1c	1.000000		
2	-1.000000	2	-0.10000
3	-1.000000	3	0.00000
4	1.000000	4	1.00000
5	1.000000	5	1.00000
6	-1.000000	6	0.00000
7	1.000000	7	1.00000
8	-1.000000	8	0.00000
9	-1.000000	9	0.00000
10	1.000000	10	0.10000
11	-1.000000	11	0.00800
12	1.000000	12	0.08000
13	1.000000	13	0.01300
14	-1.000000	14	2000.00000
15	-1.000000	15	2000.00000
16	1.000000	16	0.20000
17	-1.000000	17	0.05000
18	-1.000000	18	0.01300
19	-1.000000	19	0.75000
20	-1.000000	20	0.00000
21	1.000000	21	0.52000
22a	-1.000000	22	3.00000
22c	1.000000		
23	-1.000000	23	0.01000
24	0.677532	24	1.09357
25	1.000000	25	120.00000
26	-0.226507	26	0.04320
27	1.000000	27	0.08000
28	-1.000000	28	0.04000
29	-1.000000	29	0.02000
30	0.410575	30	61.15863
31	1.000000	31	1.20000
32	1.000000	32	0.52000
33b	1.000000	33	2.00000
33c	-1.000000		

PL-MARS model solution			
ID	**Scaled**	**ID**	**Un-scaled**
1a	-1.000000	1	3.00000
1b	-1.000000		
1c	1.000000		
2	-0.200000	2	0.30000
3	-1.000000	3	0.00000
4	-1.000000	4	0.00000
5	-1.000000	5	0.00000
6	1.000000	6	1.00000
7	-1.000000	7	0.00000
8	-1.000000	8	0.00000
9	-1.000000	9	0.00000
10	-1.000000	10	0.06000
11	-1.000000	11	0.00800
12	-1.000000	12	0.06000
13	-0.209814	13	0.00998
14	-1.00000	14	2000.00000
15	-1.000000	15	2000.00000
16	0.396026	16	0.154702
17	0.600000	17	0.25000
18	1.000000	18	0.20000
19	1.000000	19	1.50000
20	1.000000	20	15.00000
21	1.000000	21	0.52000
22a	-1.000000	22	4.00000
22c	-1.000000		
23	1.00000	23	0.01300
24	0.428571	24	1.05000
25	1.000000	25	120.00000
26	1.000000	26	0.08000
27	-1.00000	27	0.02000
28	-1.000000	28	0.04000
29	-1.000000	29	0.02000
30	-0.333332	30	50.0000
31	-1.000000	31	0.80000
32	1.000000	32	0.52000
33b	-1.000000	33	4.00000
33c	-1.000000		

Table 4.8. Objective value and solution for the first 25 constraints

No.	Name	Model used	Objective /RHS	SLR	PL-MARS
	Obj-pb05-RRS		Minimize	0.29872	0.60433
1	constr-far50-ChestD	SLR	≤ 1	0.35019	0.15127
2	constr-far50-ChestG	SLR	≤ 1	0.55169	0.53030
3	constr-far50-Chest-IP-min	PL-MARS	≥ 1	2.52737	2.76537
4	constr-far50-FemurL	PL-MARS	≤ 1	0.25288	0.41356
5	constr-far50-FemurR	PL-MARS	≤ 1	0.55534	0.44486
6	constr-far50-HIC	SLR	≤ 1	0.27121	0.63180
7	constr-far50-Head-IP-min	SLR	≥ 1	6.75682	9.12077
8	constr-far50-NeckFzMax	PL-MARS	≤ 1	0.44529	0.22826
9	constr-far50-NeckFzMin	SLR	≤ 1	0.47635	0.98620
10	constr-far50-Nij	PL-MARS	≤ 1	1.00000	0.87613
11	constr-pb05-ChestD	SLR	≤ 1	0.34780	0.22531
12	constr-pb05-ChestG	SLR	≤ 1	0.64840	0.86639
13	constr-pb05-Chest-IP-min	SLR	≥ 1	4.52105	4.11027
14	constr-pb05-FemurL	PL-MARS	≤ 1	0.20939	0.03644
15	constr-pb05-FemurR	SLR	≤ 1	0.12128	0.05257
16	constr-pb05-HIC	PL-MARS	≤ 1	0.66561	0.62946
17	constr-pb05-Head-IP-min	SLR	≥ 1	3.72812	3.97494
18	constr-pb05-NeckFzMax	SLR	≤ 1	0.34700	0.84054
19	constr-pb05-NeckFzMin	SLR	≤ 1	0.03380	0.10829
20	constr-pb05-Nij	SLR	≤ 1	0.26884	0.68289
21	constr-pb50-ChestD	PL-MARS	≤ 1	0.35458	0.38053
22	constr-pb50-ChestG	SLR	≤ 1	0.68464	1.00000
23	constr-pb50-Chest-IP-min	SLR	≥ 1	5.11609	4.28510
24	constr-pb50-FemurL	SLR	≤ 1	0.10689	0.39837
25	constr-pb50-FemurR	SLR	≤ 1	0.09968	0.26965

Table 4.9. Solution for constraints 26 to 50

No.	Name	Model used	Objective /RHS	SLR	PL-MARS
26	constr-pb50-HIC	SLR	≤ 1	0.99885	1.00000
27	constr-pb50-Head-IP-min	SLR	≥ 1	4.21463	5.31041
28	constr-pb50-NeckFzMax	SLR	≤ 1	0.28652	0.31939
29	constr-pb50-NeckFzMin	SLR	≤ 1	0.05337	0.05627
30	constr-pb50-Nij	SLR	≤ 1	0.42838	0.31123
31	constr-pu05-ChestD	SLR	≤ 1	0.32661	0.42138
32	constr-pu05-ChestG	SLR	≤ 1	0.54122	0.67659
33	constr-pu05-Chest-IP-min	SLR	≥ 1	1.00000	1.14924
34	constr-pu05-FemurL	SLR	≤ 1	0.95823	0.67842
35	constr-pu05-FemurR	SLR	≤ 1	0.99617	0.83785
36	constr-pu05-HIC	SLR	≤ 1	0.28085	0.38509
37	constr-pu05-Head-IP-min	SLR	≥ 1	5.98856	6.76919
38	constr-pu05-NeckFzMax	SLR	≤ 1	0.10464	0.19225
39	constr-pu05-NeckFzMin	SLR	≤ 1	0.31794	0.45812
40	constr-pu05-Nij	SLR	≤ 1	0.86393	0.62704
41	constr-pu50-ChestD	PL-MARS	≤ 1	0.34813	0.39099
42	constr-pu50-ChestG	SLR	≤ 1	0.74906	0.80456
43	constr-pu50-Chest-IP-min	PL-MARS	≥ 1	1.10747	1.62352
44	constr-pu50-FemurL	SLR	≤ 1	0.67282	0.91119
45	constr-pu50-FemurR	SLR	≤ 1	0.65942	0.92952
46	constr-pu50-HIC	SLR	≤ 1	1.00000	0.92596
47	constr-pu50-Head-IP-min	SLR	≥ 1	7.52112	8.95609
48	constr-pu50-NeckFzMax	SLR	≤ 1	0.12831	0.32739
49	constr-pu50-NeckFzMin	SLR	≤ 1	0.40559	0.54765
50	constr-pu50-Nij	SLR	≤ 1	0.69390	0.66288

The CPU optimization time taken by CPLEX was 0.02 seconds when using the SLR model for the objective function and 0.07 seconds when using the PL-MARS model.

4.2.2 Sensitivity Analysis and Robustness

The previous section shows an optimal solution to the MILP problem for both instances, using the SLR model and the PL-MARS model. However it can be seen that some of the values of the output variables are at the edge, i.e., equal to its corresponding right hand side value. Consequently alternatives to make the solution more robust are considered and presented in this section.

The following Table (4.10) shows the constraints that resulted equal to their RHS value.

Table 4.10. Constraints equal to their RHS values

No.	**Name**	**Model used**	**RHS**	**SLR Function Value**	**PL-MARS Function Value**
10	constr-far50-Nij	PL-MARS	≤ 1	1.00000	
33	constr-pu05-Chest-IP-min	SLR	≥ 1	1.00000	
46	constr-pu50-HIC	SLR	≤ 1	1.00000	
22	constr-pb50-ChestG	SLR	≤ 1		1.00000
26	constr-pb50-HIC	SLR	≤ 1		1.00000

A sensitivity analysis is to determine the effect that an optimal solution could have by making some changes to the original model.

The change considered in this analysis is to tighten the constraints by different ratios of the right hand side coefficients. Table 4.11 indicates the changes to the RHS values.

Table 4.11. RHS ratios

$\leq$	$\geq$
$19/20 = 0.950$	$20/19 = 1.053$
$7/8 = 0.875$	$8/7 = 1.143$
$4/5 = 0.800$	$5/4 = 1.250$
$3/4 = 0.750$	$4/3 = 1.333$

The new models obtained by tightening the constraints were solved as before, resulting in the objective function values illustrated in Figures 4.3 and 4.4, using the SLR model and the piecewise linear MARS model, respectively.

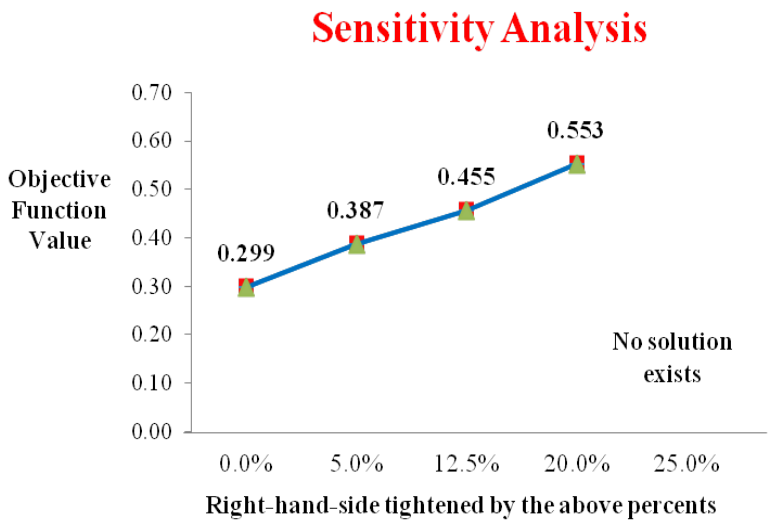

Figure 4.3. Sensitivity analysis using SLR model for the objective function.

It can be observed from Figures 4.3 and 4.4 that the objective function value increases as the tightness of the bounds increases, i.e., some depreciation occurs in the objective function values when varying the RHS coefficients, but in general does not

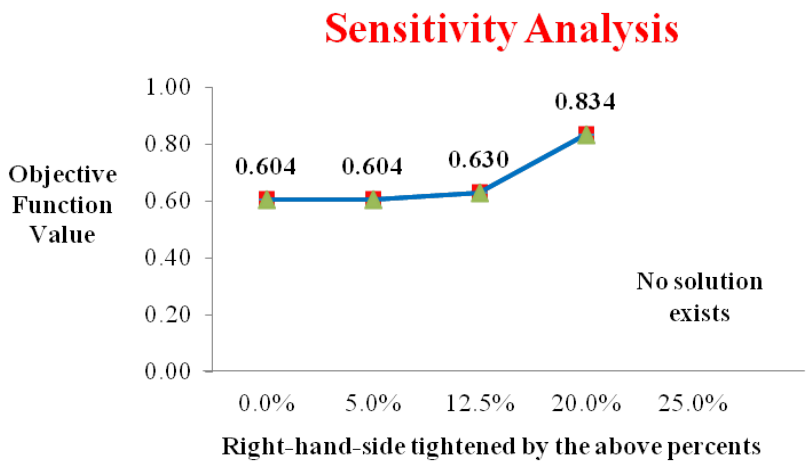

Figure 4.4. Sensitivity analysis using PL-MARS model for the objective function.

deteriorate the PL-MARS solution. No solution to the problem exists when tightening the RHS coefficients by 25%.

This analysis tested a level of robustness of the solutions, indicating a good overall performance after adding some variability to the models.

CHAPTER 5

COMPARISON AND VALIDATION

5.1 Evolutionary Algorithm Comparison

This section aims to compare the proposed method against genetic algorithms, which is a prevalent optimization technique used by researchers when combining the usage of surrogate models in regards to vehicle crashworthiness design [107–109].

5.1.1 Overview of Genetic Algorithms

Genetic algorithms (GA's) are heuristic search methods for optimization based on the principles of life. They imitate the process of natural selection and belong to the large group of evolutionary algorithms. GA's were first introduced by John Holland in 1975 ([61]) and later improved by Goldberg in 1989 ([62]).

The functionality of a traditional simple Genetic Algorithm (GA) is now explained. A GA starts by encoding the decision variables in an initial set of candidate solutions within a population; these solutions are also called genotypes, individuals, members or chromosomes. A chromosome is made of genes where every one of these units holds information and controls the inheritance of certain traits affecting future offspring. In a GA, these chromosomes are represented by a string of variables in which everyone has a feature value that would have an effect on new solutions.

Once an initial population is randomly created and evaluated by a single performance measure called fitness or evaluation function, the population starts evolving, i.e. the algorithm iteratively selects solutions and creates new generations.

This *selection* is based on the fitness measure of individuals and two genetic operators, *crossover* and *mutation*. These three are the basic elements of a GA, which are explained below along with a brief description of other control parameters used by GA's.

1) Selection strategy: the idea of this process is to select the best (most fit) individuals of the population so they can survive, and thus create new offspring, and to eliminate the individuals with the worst (least fit). Some of the most popular alternatives to determine this parameter are proportionate selection [61] and tournament selection [62].

2) Crossover: attempts to combine genes (decision variable values) from two parent chromosomes (existing solutions) and creates a new solution by swapping certain segments of the parents' strings. Crossover probability indicates the rate in which crossover is performed. If this probability is set at a 0%, the new generation would be an exact copy of chromosomes from previous population (parents). The most common crossover operators are one-point [61], and uniform crossover [110].

3) Mutation: changes randomly one or more genes of the chromosomes (individuals) of the current population, creating a new candidate solution. Mutation probability controls how often genes of a chromosome will be mutated. If this probability is to be selected at a high level, too many genes would be randomly changed and the new chromosomes would have nothing in common with its parents; otherwise if no mutation is performed, nothing would be changed. Two of the most common alternative mutation method is uniform mutation and non-uniform mutation.

4) Population size: indicates the number of chromosomes in a population, i.e. in one generation. If the size of the population contains a very few number of individuals, the algorithm may only explore a small part of the search space, and thus converge

too quickly. However if this number is set too high, it may be the case that some GA algorithms present computational time issues, slowing down the process.

5) Encoding of chromosomes: this has to do with the way to create or represent the chromosomes. Holland's GA establishes the string of chromosomes by using ones and zeros, or 'bits', however binary encoding is not the only option; other alternatives are: permutation encoding, tree encoding, and value encoding [111].

6) Generation size: this parameter determines the number of iterations the GA algorithm has to perform. If this parameter is set too small, the search space may not be completely explored.

As it can be noticed, there are many GA variations, and thus many possible combinations to set these parameters. While there is no universal best method to set such parameters for any problem, this study is limited to a particular simple GA, which was built by Denis Cormier (North Carolina State University) and modified by Sita S. Raghavan (University of North Carolina at Charlotte). This GA code is available from Michalewicz [112].

Its corresponding pseudo code is presented in Algorithm 1.

The following is a quote from Michalewicz [112] "Each generation involves selecting the best members, performing crossover and mutation and then evaluating the resulting population" this is performed until the terminating condition is satisfied, i.e. the iterative process achieves the maximum number of generations (MAXGENS).

This simple GA assumes there is no distinction between the fitness of an individual and the value objective function. It uses proportional selection, one-point crossover, uniform mutations, and includes a routine called elitist, which ensures that the best chromosomes are retained between generations.

Algorithm 1 Genetic Algorithm pseudo code by Cormier and Raghavan [112]

INITIALIZE: initializes the genes within the variables bounds

EVALUATE: implements the user-defined valuation function

KEEP THE BEST: keeps track of the best member of the population

while generation < MAXGENS **do**

SELECTOR performs standard proportional selection

CROSSOVER: selects two parents for the single point crossover

MUTATE: performs a random uniform mutation

EVALUATE: implements the user-defined valuation function

ELITIST: stores the best member of the previous generation

end while

5.1.2 Customization of Genetic Algorithm

The encoding type used for the chromosomes was based on value encoding, i.e. every solution is a string of some values. For the problem being analyzed, this means that every solution consists of 37 different values, which represent the explanatory variables of the crash safety system design case study. The lower and upper bounds of these variables are set to -1 and +1, respectively. Note that this will only give real numbers to the variables, and there are 14 binary variables, and a need for three packing constraints, therefore a new encoding system is employed.

Let τ represent a threshold used to determine the binary value (-1 or +1) for the corresponding variables, and n symbolize the number of levels of the categorical variable (2, 3, and 4 are the levels this problem requires).

$$\tau = \frac{2}{\sqrt[n-1]{n}} - 1 \tag{5.1}$$

Here is an example of how this threshold works; first, the GA algorithm randomly generates values between -1 and +1 for each gene (variable) of each chromosome (candidate solution) in the population. Then considering as an example variable 1 which has 4 levels, the value for τ would be:

$$\tau = \frac{2}{\sqrt[4-1]{4}} - 1 = 0.2599 \tag{5.2}$$

Now, if the maximum fractional value of the variables representing the levels of variable 1, that is 1a, 1b, or 1c is $\geq \tau$ then the value for this variable is rounded to +1 and the rest of them become -1. If the maximum value of these variables is not $\geq \tau$ then all of them are rounded to a -1 value, indicating that the absent variable is the selected value. The pseudo code for τ is presented in Algorithm 2.

Algorithm 2 Pseudo code for τ

τ for categorical variable j is previously computed based on number of levels n.

for all $j \in K$ **do**

Find $\bar{l} \in \arg\max(x_l \in L(j))$

$\forall l \in L(j) \setminus \{\bar{l}\}$, set $x_l = -1$

if $x_{\bar{l}} \geq \tau$ **then**

$x_{\bar{l}} = +1$

else

$x_{\bar{l}} = -1$

end if

end for

This rounding approach is applied for all the binary variables with n ≥ 2 levels. Notice that the threshold value for the variables with only 2 categorical levels is equal to zero. With such rounding encoding, the packing constraints are implicit.

The fitness value of each member of the population is then calculated with the vector of variables that includes the rounded solutions.

As indicated in Algorithm 1, the evaluation function is implemented based on the user needs, in this case the fitness function is either the SLR model or the PL-MARS model. However the case study under analysis is subject to 50 constraint functions that need to be accounted for in the algorithm.

A GA algorithm in basically an unconstrained search method, however constraints can be incorporated to the evaluation function by the presence of a moderate penalty applied to all of the violated constraints.

A constraint function is violated if does not satisfy equation 4.2
If this is the case, a violating value ν_i is obtained by the following equation:

$$\nu_i = [\hat{g}_i(x) - b_i]_+, \qquad \forall i = 2, ..., 51, \tag{5.3}$$

The sum of all the violations is then represented by:

$$\Delta = \sum_{i=2}^{G} \nu_i \tag{5.4}$$

Finally Δ is multiplied by a penalty value.
The customized GA is now referred to as: PL-MARS-GA.

5.1.3 PL-MARS-GA Computational Results

Using the GA presented in Algorithm 1 with the implementation of the variations discussed above, the PL-MARS-GA runs were performed using the two different GA parameter settings, presented in Table 5.1.

Table 5.1. GA parameter settings

Parameters	**Grefenstette [113]**	**Cormier and Raghavan [112]**
Population size	30	50
Maximum number of generations	300	1000
Probability of crossover	0.9	0.8
Probability of mutation	0.01	0.15

Grefenstette (G) in 1986 [113] conducted some experiments for searching and determining optimal control parameters for a class of global optimization procedures, suggesting the values shown in second column. The third column shows the set of parameter settings used for the simple GA proposed by Cormier and Raghaven (C&R) [112].

Ten trials for each of the two objective function models (i.e. the SLR model and the PL-MARS models) using the parameters shown in Table 5.1 (5 with G-settings and 5 with C&R-settings) were performed. Different penalty values were applied to each run. These trials were then run tightening the RHS by 0%, 5%, 12.5% and 20% respectively.

A C-programming code executed on a Dual 2.6 GHz Athlon workstation was used to generate the results given in Tables 5.2 - 5.5, where the last two columns (V.C. which stands for violated constraints and SumV which stands for the sum of the violation) indicate the number of constraints violated (if any) and by how much. Table 5.2 includes the PL-MARS-GA runs when the rhs is tightened by 0% and 5%

using the two objective function models with penalty values of 5, 10, 20, 30, and 50; and employing the G and C&R parameter settings. Table 5.3 shows the PL-MARS-GA runs using the same settings as in previous table but when the rhs is tightened by 12.5% and 20%. Table 5.4 includes the PL-MARS-GA runs when the rhs is tightened by 0% and 5% using the two objective function models with penalty values of .01, .05, .10, .20, and .30; and employing the G and C&R parameter settings. Finally, Table 5.5 shows the PL-MARS-GA runs using the same settings as in previous table but when the rhs is tigtened by 12.5%.

As a result of these runs, where 130 PL-MARS-GA points were obtained, several conclusions can be made; the PL-MARS-GA runs showing better performance are those in which the penalty is set to a higher value, from Tables 5.4 and 5.5 it can be seen how bad the constraints were violated. This is because when the penalty values are set too small, the GA generates many infeasible candidate solutions, and thus violated constraints. Overall the C&R-settings show better results than the G-settings. The PL-MARS-GA algorithm was capable of finding only one feasible solution when the constraints were tightened by a 20%.

All the trials were performed within a reasonable amount of time, less than 1 second and 2 seconds for the G and C&R settings respectively.

A clear disadvantage of any evolutionary algorithm is that there is no certainty that the solution found is an optimal solution, and thus is the case for the PL-MARS-GA feasible points.

5.2 Validation Procedure

Since it is assumed that there is no access to a crash simulator and experiments are expensive, other alternative methods to validate the solutions obtained

Table 5.2. PL-MARS-GA runs with rhs tightened by 0% and 5%, and penalty values of 5, 10, 20, 30 and 50.

Point	Model used	Tighten of rhs	Penalty	Settings	Objective function	Constraints	
						V.C.	SumV
1	PL-MARS	0%	5	C&R	0.638816	0	0
2	PL-MARS	0%	10	C&R	0.624652	0	0
3	PL-MARS	0%	20	C&R	0.623851	0	0
4	PL-MARS	0%	30	C&R	0.616506	0	0
5	PL-MARS	0%	50	C&R	0.624841	0	0
6	PL-MARS	0%	5	G	0.659368	0	0
7	PL-MARS	0%	10	G	0.699837	0	0
8	PL-MARS	0%	20	G	0.693529	0	0
9	PL-MARS	0%	30	G	0.699266	0	0
10	PL-MARS	0%	50	G	0.71575	0	0
11	SLR	0%	5	C&R	0.39904	1	0.002510
12	SLR	0%	10	C&R	0.387498	0	0
13	SLR	0%	20	C&R	0.438837	0	0
14	SLR	0%	30	C&R	0.454646	0	0
15	SLR	0%	50	C&R	0.473381	0	0
16	SLR	0%	5	G	0.506686	0	0
17	SLR	0%	10	G	0.506157	0	0
18	SLR	0%	20	G	0.495974	0	0
19	SLR	0%	30	G	0.495974	0	0
20	SLR	0%	50	G	0.485137	0	0
21	PL-MARS	5%	5	C&R	0.614535	0	0
22	PL-MARS	5%	10	C&R	0.624843	0	0
23	PL-MARS	5%	20	C&R	0.667346	0	0
24	PL-MARS	5%	30	C&R	0.648628	0	0
25	PL-MARS	5%	50	C&R	0.642663	0	0
26	PL-MARS	5%	5	G	0.81637	0	0
27	PL-MARS	5%	10	G	0.731882	0	0
28	PL-MARS	5%	20	G	0.732541	0	0
29	PL-MARS	5%	30	G	0.76729	0	0
30	PL-MARS	5%	50	G	0.772504	0	0
31	SLR	5%	5	C&R	0.481685	0	0
32	SLR	5%	10	C&R	0.456689	0	0
33	SLR	5%	20	C&R	0.462120	0	0
34	SLR	5%	30	C&R	0.499993	0	0
35	SLR	5%	50	C&R	0.489117	0	0
36	SLR	5%	5	G	0.631407	0	0
37	SLR	5%	10	G	0.559625	0	0
38	SLR	5%	20	G	0.608929	0	0
39	SLR	5%	30	G	0.655314	0	0
40	SLR	5%	50	G	0.65314	0	0

Table 5.3. PL-MARS-GA runs with rhs tightened by 12.5% and 20%, and penalty values of 5, 10, 20, 30 and 50.

Point	Model used	Tighten of rhs	Penalty	Settings	Objective function	Constraints	
						V.C.	SumV
41	PL-MARS	12.5%	5	C&R	0.724342	0	0
42	PL-MARS	12.5%	10	C&R	0.744787	0	0
43	PL-MARS	12.5%	20	C&R	0.751777	0	0
44	PL-MARS	12.5%	30	C&R	0.772278	0	0
45	PL-MARS	12.5%	50	C&R	0.760643	0	0
46	PL-MARS	12.5%	5	G	0.823934	0	0
47	PL-MARS	12.5%	10	G	0.910547	0	0
48	PL-MARS	12.5%	20	G	0.852484	0	0
49	PL-MARS	12.5%	30	G	0.814940	0	0
50	PL-MARS	12.5%	50	G	0.904565	0	0
51	SLR	12.5%	5	C&R	0.564163	0	0
52	SLR	12.5%	10	C&R	0.611356	0	0
53	SLR	12.5%	20	C&R	0.552554	0	0
54	SLR	12.5%	30	C&R	0.571005	0	0
55	SLR	12.5%	50	C&R	0.538343	0	0
56	SLR	12.5%	5	G	0.59676	0	0
57	SLR	12.5%	10	G	0.647900	0	0
58	SLR	12.5%	20	G	0.552341	0	0
59	SLR	12.5%	30	G	0.552341	0	0
60	SLR	12.5%	50	G	0.552341	0	0
61	PL-MARS	20%	5	C&R	1.028585	2	0.167959
62	PL-MARS	20%	10	C&R	1.015177	2	0.094728
63	PL-MARS	20%	20	C&R	0.985572	0	0
64	PL-MARS	20%	30	C&R	1.051180	2	0.093177
65	PL-MARS	20%	50	C&R	1.713455	1	0.733330
66	PL-MARS	20%	5	G	1.094850	2	0.186001
67	PL-MARS	20%	10	G	1.492022	2	0.568149
68	PL-MARS	20%	20	G	2.017517	3	1.176426
69	PL-MARS	20%	30	G	2.545073	2	1.621200
70	PL-MARS	20%	50	G	4.739970	2	3.868085
71	SLR	20%	5	C&R	0.718831	2	0.027693
72	SLR	20%	10	C&R	0.963811	2	0.190985
73	SLR	20%	20	C&R	0.779890	2	0.063150
74	SLR	20%	30	C&R	0.969601	2	0.251800
75	SLR	20%	50	C&R	0.896558	2	0.198434
76	SLR	20%	5	G	0.854567	2	0.191709
77	SLR	20%	10	G	1.322959	1	0.590329
78	SLR	20%	20	G	1.887270	2	1.082857
79	SLR	20%	30	G	2.970729	3	2.180923
80	SLR	20%	50	G	3.691352	2	2.894758

Table 5.4. PL-MARS-GA runs with rhs tightened by 0% and 5%, and penalty values of .01, .05, .10, .20 and .30.

Point	Model used	Tighten of rhs	Penalty	Settings	Objective function	Constraints	
						V.C.	SumV
81	PL-MARS	0%	.01	G	0.627153	0	0
82	PL-MARS	0%	.05	G	0.663975	1	0.005773
83	PL-MARS	0%	.10	G	0.637587	0	0
84	PL-MARS	0%	.20	G	0.641184	0	0
85	PL-MARS	0%	.30	G	0.648476	0	0
86	PL-MARS	0%	.01	C&R	0.606184	0	0
87	PL-MARS	0%	.05	C&R	0.608583	0	0
88	PL-MARS	0%	.10	C&R	0.613622	0	0
89	PL-MARS	0%	.20	C&R	0.617395	0	0
90	PL-MARS	0%	.30	C&R	0.635638	0	0
91	SLR	0%	.01	G	0.296273	4	0.012950
92	SLR	0%	.05	G	0.400476	5	0.080722
93	SLR	0%	.10	G	0.406014	1	0.025561
94	SLR	0%	.20	G	0.414472	0	0
95	SLR	0%	.30	G	0.505433	0	0
96	SLR	0%	.01	C&R	0.271191	4	0.013855
97	SLR	0%	.05	C&R	0.325506	4	0.044389
98	SLR	0%	.10	C&R	0.366913	2	0.013021
99	SLR	0%	.20	C&R	0.362006	1	0.003072
100	SLR	0%	.30	C&R	0.382291	1	0.003982
101	PL-MARS	5%	.01	G	0.637651	5	0.000991
102	PL-MARS	5%	.05	G	0.643357	1	0.001961
103	PL-MARS	5%	.10	G	0.658087	2	0.001930
104	PL-MARS	5%	.20	G	0.667829	0	0
105	PL-MARS	5%	.30	G	0.632253	0	0
106	PL-MARS	5%	.01	C&R	0.609757	0	0
107	PL-MARS	5%	.05	C&R	0.626556	0	0
108	PL-MARS	5%	.10	C&R	0.622408	0	0
109	PL-MARS	5%	.20	C&R	0.61849	0	0
110	PL-MARS	5%	.30	C&R	0.627622	0	0
111	SLR	5%	.01	G	0.392673	5	0.011861
112	SLR	5%	.05	G	0.389013	5	0.045029
113	SLR	5%	.10	G	0.410777	4	0.054217
114	SLR	5%	.20	G	0.474167	0	0
115	SLR	5%	.30	G	0.459844	3	0.024705
116	SLR	5%	.01	C&R	0.253594	7	0.014794
117	SLR	5%	.05	C&R	0.313015	6	0.056350
118	SLR	5%	.10	C&R	0.357937	3	0.005158
119	SLR	5%	.20	C&R	0.402634	2	0.006409
120	SLR	5%	.30	C&R	0.438045	3	0.025950

Table 5.5. PL-MARS-GA runs with rhs tightened by 12.5%, and penalty values of .01, .05, .10, .20 and .30.

Point	Model used	Tighten of rhs	Penalty	Settings	Objective function	Constraints	
						V.C.	SumV
121	PL-MARS	12.5%	.01	G	0.639894	3	0.002683
122	PL-MARS	12.5%	.05	G	0.643248	2	0.002631
123	PL-MARS	12.5%	.10	G	0.659409	3	0.015052
124	PL-MARS	12.5%	.20	G	0.665173	2	0.006911
125	PL-MARS	12.5%	.30	G	0.683692	2	0.018274
126	PL-MARS	12.5%	.01	C&R	0.628356	2	0.000295
127	PL-MARS	12.5%	.05	C&R	0.626324	2	0.003548
128	PL-MARS	12.5%	.10	C&R	0.624719	3	0.005737
129	PL-MARS	12.5%	.20	C&R	0.640644	2	0.006434
130	PL-MARS	12.5%	.30	C&R	0.65294	3	0.019983

from different sources have to be studied, and thus the following validation procedure is conducted.

Considering the two initial solutions presented in Table 4.7, which were found by applying the proposed method (now referred to as MARSOPT), and the ones generated by tightening the constraints, there are a total of 8 MARSOPT solutions (4 obtained using a SLR model to approximate the objective function and 4 using a PL-MARS model); there are also 130 PL-MARS-GA solutions (60 obtained using a SLR model to approximate the objective function and 70 using a PL-MARS model), and a total of 1449 initial points (200 from Data set 1 and 1249 from Data set 2). All these solutions are examined by six evaluators, where two of them are related to the objective function values and the other four deal with the robustness of feasibility (i.e., the constraint functions created with the approximation models). However, it should be mentioned that these are predicted evaluators since the values obtained are based on the approximation models built for the objective function and the constraints.

Two objective functions were generated, one approximated by an SLR model and the other by a PL-MARS model (please refer to subsection 4.2.1). Each solution is plugged into these functions and an objective function value for each is computed. Now, each of the solutions is as well plugged into all of the constraint functions in order to determine how many of these constraints are violated (if any) and which of them are within 1%, 5% and 10% at the edge of their corresponding bounds.

The results of this analysis is given in Table 5.6 and for simplicity purposes the rest of the results are partially given in Tables 5.8 - 5.7.

Table 5.6. MARSOPT points

Solutions				**Evaluators**					
	Model	**Tighten**	**Objective**	**Objective function**		**Constraints**			
Point	**used**	**of rhs**	**function**	**SLR**	**PL-MARS**	**V.C.**	**1%**	**5%**	**10%**
1	SLR	0%	0.298722	0.298722	0.776678	0	2	3	3
2	SLR	5%	0.386710	0.386710	0.611566	0	0	3	4
3	SLR	12.5%	0.455111	0.455111	0.772666	0	0	0	0
4	SLR	20%	0.552853	0.552853	0.993936	0	0	0	0
5	PL-MARS	0%	0.604331	0.699816	0.604331	0	0	1	4
6	PL-MARS	5%	0.604331	0.564830	0.604331	0	0	2	4
7	PL-MARS	12.5%	0.629895	0.745441	0.629896	0	0	0	0
8	PL-MARS	20%	0.834261	0.834241	0.834262	0	0	0	0

Table 5.7. PL-MARS-GA points

Solutions						Evaluators					
						Objective function		Constraints			
Point	Model used	Tighten of rhs	Penalty	Settings	Objective function	SLR	PL-MARS	V.	1%	5%	10%
1	PL-MARS	0%	5	C&R	0.638816	0.58977	0.638816	0	0	2	4
2	PL-MARS	0%	10	C&R	0.624652	0.73701	0.624652	0	0	2	4
3	PL-MARS	0%	20	C&R	0.623851	0.66343	0.623851	0	0	1	3
4	PL-MARS	0%	30	C&R	0.616506	0.65445	0.616506	0	0	2	4
5	PL-MARS	0%	50	C&R	0.624841	0.68119	0.624841	0	1	2	2
:											
126	PL-MARS	12.5%	.01	C&R	0.628356	0.770272	0.628061	2	0	0	1
127	PL-MARS	12.5%	.05	C&R	0.626324	0.741732	0.622776	2	0	0	1
128	PL-MARS	12.5%	.10	C&R	0.624719	0.756682	0.618982	3	0	0	1
129	PL-MARS	12.5%	.20	C&R	0.640644	0.791965	0.634210	2	0	0	1
130	PL-MARS	12.5%	.30	C&R	0.65294	0.806137	0.632957	3	0	0	1

Table 5.8. Data Set 1

Solutions		**Evaluators**					
	Objective	**Objective function**		**Constraints**			
Points	**function**	**SLR**	**PL-MARS**	**V.C.**	**1%**	**5%**	**10%**
1	0.588700	0.589964	0.749453	2	0	2	4
2	0.615278	0.641593	0.653461	1	0	0	2
3	0.617033	0.715519	0.691394	4	0	0	1
4	0.847828	0.691839	0.858577	1	0	0	2
:							
198	1.365967	1.317599	1.343026	5	0	0	4
199	0.573086	0.618299	0.604331	0	0	1	4
200	1.365967	1.313159	1.343026	5	0	0	3

Table 5.9. Data Set 2

Solutions		**Evaluators**					
	Objective	**Objective function**		**Constraints**			
Points	**function**	**SLR**	**PL-MARS**	**V.C.**	**1%**	**5%**	**10%**
1	0.898130	1.318813	1.270186	10	0	2	6
2	1.504252	0.914360	0.937300	5	0	0	3
3	0.773490	1.385123	1.459975	11	0	0	1
4	0.689225	1.045283	1.041535	10	0	2	5
:							
1247	0.624069	0.605319	0.604331	0	0	1	3
1248	0.644536	0.647973	0.653461	0	0	1	3
1249	0.617241	0.605319	0.604331	0	0	1	4

After obtaining the values for all of the points, a Pareto optimal analysis with respect to the six evaluators in needed, in which all of them are to be minimized. Pareto optimality is a measure of efficiency [1] that in this case, seeks for the solutions that are better in all six evaluators and cannot be improved without deteriorating any of the other evaluators. These solutions are then called non-dominated solutions.

[1] *http : //www.gametheory.net*

In this analysis only the feasible points are considered, that is all of the solutions with at least one violated constraint, are disregarded. Tables 5.10 and 5.11 contain the 27 solutions that are not strictly dominated by any other solution, and thus are on the efficient frontier.

Table 5.10. Efficient Pareto frontier points

Solutions							Evaluators					
							Objective function		Constraints			
Point	**Model used**	**Source**	**Tighten of rhs**	**Penalty**	**Settings**	**Objective function**	**SLR**	**PL-MARS**	**V.**	**1%**	**5%**	**10%**
3	SLR	MARSOPT	12%			0.455111	0.455111	0.772666	0	0	0	0
903		Data set 2				0.656917	0.641593	0.653461	0	0	0	0
7	PL-MARS	MARSOPT	12%			0.629895	0.745441	0.629896	0	0	0	0
467		Data set 2				0.628998	0.603073	0.653461	0	0	0	1
1180		Data set 2				0.606366	0.603073	0.653461	0	0	0	1
1199		Data set 2				0.606366	0.603073	0.653461	0	0	0	1
25	PL-MARS	PL-MARS-GA	5%	50	C&R	0.642663	0.715846	0.642663	0	0	0	1
32	SLR	PL-MARS-GA	5%	10	C&R	0.456689	0.456689	0.724496	0	0	0	2
33	SLR	PL-MARS-GA	5%	20	C&R	0.462120	0.462120	0.690006	0	0	0	2
1223		Data set 2				0.606366	0.583973	0.653461	0	0	0	2
107	PL-MARS	PL-MARS-GA	5%	.05	C&R	0.626556	0.772881	0.626556	0	0	0	2
1140		Data set 2				0.685418	0.549753	0.653461	0	0	0	3
25		Data set 1				0.600822	0.594379	0.604331	0	0	0	3
1244		Data set 2				0.593866	0.594379	0.604331	0	0	0	3

Table 5.11. Efficient Pareto frontier points - cont.

Solutions							Evaluators					
							Objective function		Constraints			
Point	Model used	Source	Tighten of rhs	Penalty	Settings	Objective function	SLR	PL-MARS	V.	1%	5%	10%
1244		Data set 2				0.593866	0.594379	0.604331	0	0	0	3
137		Data set 1				0.574414	0.586729	0.604331	0	0	0	4
114	SLR	PL-MARS-GA	5%	.20	G	0.474167	0.474167	0.682065	0	0	0	5
301		Data set 2				0.611779	0.560213	0.653461	0	0	1	2
311		Data set 2				0.611779	0.560213	0.653461	0	0	1	2
1203		Data set 2				0.617241	0.590999	0.604331	0	0	1	3
152		Data set 1				0.573086	0.571139	0.604331	0	0	1	4
94		Data set 1				0.574414	0.574999	0.604331	0	0	2	3
12	SLR	PL-MARS-GA	0%	10	C&R	0.387498	0.387498	0.683940	0	0	2	4
452		Data set 2				0.599518	0.538593	0.653461	0	0	2	4
6	PL-MARS	MARSOPT	5%			0.604331	0.564830	0.604331	0	0	2	4
2	SLR	MARSOPT	5%			0.386710	0.386710	0.611566	0	0	3	4
5	PL-MARS	PL-MARS-GA	0%	50	C&R	0.624841	0.681191	0.624841	0	1	2	2
1	SLR	MARSOPT	0%			0.298722	0.298722	0.776678	0	2	3	3

Table 5.12 summarizes the total amount of points from different sources that were evaluated (1587), from which only 456 of them were feasible solutions. It also indicates the number of points that used a SLR model to approximate the objective function and the number of points in which the objective was modeled with a PL-MARS function. Last column shows the total number of Pareto optimal points (27), resulting 5 out of the 8 total MARSOPT points on the efficient frontier. Less than 6% of the PL-MARS-GA points appear on the efficient frontier and only 2% and less that 1% of the points come from the Data sets 1 and 2, respectively.

Table 5.12. Summary of points

Points	**Algorithm used/source**	**Model used**	**Pareto optimal points**
4	MARSOPT	SLR	3
4	MARSOPT	PL-MARS	2
60	PL-MARS-GA	SLR	4
70	PL-MARS-GA	PL-MARS	3
200	Data set 1		4
1249	Data set 2		11

Table 5.13 shows how the other three MARSOPT points were dominated.

Table 5.13. Dominated MARSOPT points

Solutions					Evaluators					
					Objective function		Constraints			
Point	Model used	Algorithm used/source	Tighten of rhs	Objective function	SLR	PL-MARS	V.	1%	5%	10%
This point:										
4	SLR	MARSOPT	20%	0.552853	0.552853	0.993936	0	0	0	0
was dominated by:										
3	SLR	MARSOPT	12.5%	0.455111	0.45111	0.772666	0	0	0	0
This point:										
8	PL-MARS	MARSOPT	20%	0.834261	0.834241	0.834262	0	0	0	0
was dominated by:										
3	SLR	MARSOPT	12.5%	0.455111	0.45111	0.772666	0	0	0	0
7	PL-MARS	MARSOPT	12.5%	0.629895	0.745441	0.629896	0	0	0	0
This point:										
5	PL-MARS	MARSOPT	0%	0.604331	0.699816	0.604331	0	0	1	4
was dominated by:										
25		Data set 1		0.600822	0.594379	0.604331	0	0	0	3
1244		Data set 2		0.593866	0.594379	0.604331	0	0	0	3
137		Data set 1		0.574414	0.586729	0.604331	0	0	0	4
1203		Data set 2		0.617241	0.590999	0.604331	0	0	1	3
152		Data set 1		0.573086	0.571139	0.604331	0	0	1	4

The un-scaled vector solutions for the five MARSOPT points that resulted on the efficient frontier are demostrated in Table 5.14 along with the vector solution of the point number 152 from the Data set 1, which is in fact one of the best points from this set with respect to the true objective function value; 33 out of the 200 points have the same value (0.573086) but only point 152 resulted on the efficient frontier. Table 5.15 shows the model used and the solution obtained for the objective function by using the proposed method (MARSOPT points) and the true objective function value for the point number 152. It also includes the values of the evaluators related to the objective function using the SLR and PL-MARS models to approximate the objective function respectively.

Although this study has been demonstrating results using two models to represent the objective function, piecewise linear MARS models are of main interest and the initial focus of this research. Therefore the subsequent conclusion is made only for the solution points obtained when using a piecewise linear MARS model to approximate the objective function (MARSOPT points 6 and 7).

It can be seen from Table 5.14 that some of the variable values are highlighted. These variables in fact are the ones that appear in the piecewise linear MARS objective function (variables 9, 14, 24, 30 and 32). If these values are compared to the corresponding variables of the point number 152 from the Data set 1, very few differences between them can be observed. In fact MARSOPT point number 6 has the exact same values.

Finally Table 5.16 shows the requirement that was given along with the original data in order to determine if the resulting Relative Risk Score Star Rating (RRS) would meet a 4 or 5 star rating.

Table 5.14. Un-scaled solutions for selected points

	MARSOPT					Data set 1
ID	**Point 1**	**Point 2**	**Point 3**	**Point 6**	**Point 7**	**Point 152**
1	3	3	3	3	3	4
2	-0.1	0.33247	0.427286	0.3	0.427286	0.1
3	0	0	0	0	0	0
4	1	1	1	0	1	1
5	1	1	1	0	1	1
6	0	0	0	1	0	0
7	1	0	0	0	0	0
8	0	0	0	0	0	0
9	0	0	0	0	0	0
10	0.100	0.100	0.100	0.100	0.100	0.100
11	0.008	0.008	0.008	0.013	0.008	0.013
12	0.080	0.080	0.060	0.080	0.060	0.080
13	0.013	0.013	0.013	0.008	0.013	0.013
14	2000	2000	2000	2000	2000	2000
15	2000	2000	2000	2000	2000	2800
16	0.200	0.200	0.200	0.176	0.200	0.150
17	0.050	0.050	0.050	0.250	0.050	0.250
18	0.013	0.013	0.013	0.013	0.013	0.013
19	0.750	0.750	0.750	0.750	0.750	1.000
20	0.000	0.000	0.000	0.940	0.000	0.000
21	0.520	0.520	0.520	0.520	0.520	0.520
22	3	3	3	1	3	3
23	0.010	0.010	0.010	0.010	0.010	0.010
24	1.094	1.050	1.009	1.050	1.009	1.050
25	120	120	120	120	120	60
26	0.043	0.080	0.080	0.080	0.080	0.060
27	0.080	0.020	0.020	0.080	0.020	0.060
28	0.040	0.040	0.040	0.040	0.040	0.080
29	0.020	0.040	0.020	0.020	0.020	0.020
30	61.159	51.759	44.945	50.000	44.945	50.000
31	1.200	0.800	0.800	0.800	0.800	1.200
32	0.520	0.520	0.520	0.520	0.520	0.520
33	2	2	2	4	2	4

Table 5.15. Objective function values for selected points

	Solutions		Obj. fun. evaluators	
Point	**Model used**	**Obj. fun.**	**SLR**	**PL-MARS**
MARSOPT 1	SLR	0.298722	0.298722	0.776678
MARSOPT 2	SLR	0.386710	0.386710	0.611566
MARSOPT 3	SLR	0.455111	0.455111	0.772666
MARSOPT 6	PL-MARS	0.604331	0.564830	0.604331
MARSOPT 7	PL-MARS	0.629895	0.745441	0.629896
Data set 1 152		0.573086	0.571139	0.604331

Table 5.16. Relative Risk Score Star Rating

RRS $\leq$ 0.67	5 Stars
0.67$\leq$ RRS $<$ 1.33	4 Stars
1.33$\leq$ RRS $<$ 2.00	3 Stars
2.00$\leq$ RRS $<$ 2.67	2 Stars
RRS $>$ 2.67	1 Star

Table 5.17. MARSOPT points - RRS Star Rating

Solutions				Evaluators	
				Objective function	
Point	**Model used**	**Tighten of rhs**	**Objective function**	**SLR**	**PL-MARS**
1	SLR	0%	0.298722	0.298722	0.776678
2	SLR	5%	0.386710	0.386710	0.611566
3	SLR	12.5%	0.455111	0.455111	0.772666
4	SLR	20%	0.552853	0.552853	0.993936
5	PL-MARS	0%	0.604331	0.699816	0.604331
6	PL-MARS	5%	0.604331	0.564830	0.604331
7	PL-MARS	12.5%	0.629895	0.745441	0.629896
8	PL-MARS	20%	0.834261	0.834241	0.834262

It can be concluded that a 5-star rating was achieved with 7 of the MARSOPT solutions obtained, and one achieved a 4-star rating (see the 'Objective function' column in Table 5.17). However it should be mentioned that these are predicted ratings based on the surrogate models that were globally optimized.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 Concluding Remarks

This research presented a novel deterministic global optimization method based on mixed integer linear programming to optimize piecewise linear functions generated by a modified version of multivariate adaptive regression splines (MARS). The method is computationally fast and is also capable of handling non-convexity, non-linearity and is adjusted to manage continuous and categorical variables.

This method was applied to search for an optimal solution of two case studies. The first one was to optimize a single piecewise linear MARS function of a non-convex minimization inventory test problem, while the second case was to minimize the crash performance of a vehicle safety system design example that consisted of 33 design variables and 50 output variables.

The method was able to globally optimize the surrogate models representing the search space of the problems, where stepwise linear regression and piecewise linear MARS models were approximated. These meta-models were built from a relatively small set of design variables.

For the second case study, the method was also effective in providing a more reliable robust design, demonstrating a very small deterioration on the solutions when the robustness was evaluated by tightening the constraints.

Also, while the main drawback of problems such as the vehicle crashworthiness design is the requirement of an extensive amount of computational time, this method was able to give an optimal solution in less than a second.

The proposed method was compared to a customized genetic algorithm, which was altered with a penalty applied to the evaluation function in order to account for the output variables. Although this evolutionary algorithm was able to provide some feasible solutions, the main disadvantage of it is that there is no a guaranty that the solutions found optimize the surrogate model.

A validation procedure based on six evaluators was also executed, in which as a result, 27 of all the available solutions resulted on the efficient frontier, including 5 out of the 8 MARSOPT points.

A relative risk score, with a 5-star rating as the highest, was given along with the data of the second case study in order to determine the star rating of the results, for which 7 of the MARSOPT points achieved a 5-star rating, while the other point achieved a 4-star rating.

6.2 Future Work

Even though the strength of the proposed method can be seen by the success in identifying Pareto optimal solutions, there is an important factor that can be considered as an improvement.

When there exists correlation between two explanatory variables, a meta-model maintains the one with the least amount of variation as a better predictor and drops the other one. This is very good when building only one approximation model. However in the implementation of the proposed method when a piecewise linear MARS model is subject to other linear models, the correlation of these variables should be considered closely.

If assuming that a highly correlated explanatory variable is dropped from the model used to approximate the objective function, which later appears to be a sig-

nificant variable in another model used to approximate one of the output variables, separating the effect of these two variables may not be beneficial from an optimization point of view, since a variable that may have altered the objective function value could have been ignored.

Therefore an alternative approach is to consider incorporating constraints for these variables and ensuring their appearance as significant control variables.

Correlation coefficient analysis can be performed to determine the correlation strength between the variables so they can be considered in the optimization formulation. Also, a variance inflation factor, which measures how much the variance can be affected due to relationship between two or more explanatory variables, can be used.

In general, not only the correlation between the variables within a model should be considered but also between the models.

This previous reason may explain why the comparison of Table 5.14 showed exact results for only the variables that appeared in the objective function model.

Along with the previous suggestion, another important consideration for future applications is to control, whenever possible, the way the data are collected in order to ensure more successful results.

Finally, the proposed method can be generalized by applying it to other large-scale optimization problems.

APPENDIX A

Mir-Range and Half-Range Values

In this appendix, the mid-range and half-range formulas and values used are presented. Then, the formulas to convert a scaled variable value to an un-scaled and vice versa are given.

$$midrange = \frac{min + max}{2}, \tag{A.1}$$

$$halfrange = \frac{max - min}{2} \tag{A.2}$$

where *min* and *max* represent the minimum and maximum values of each un-scaled variable respectively.

Table A.1. Mid-range & Half-range values

ID	**mid-range**	**half-range**	**ID**	**mid-range**	**half-range**
1a	0.0000	1.0000	18	0.1065	0.0935
1b	0.0000	1.0000	19	1.1250	0.3750
1c	0.0000	1.0000	20	7.5000	7.5000
2	0.4000	0.5000	21	0.4600	0.0600
3	0.0000	1.0000	22a	0.0000	1.0000
4	0.0000	1.0000	22c	0.0000	1.0000
5	0.0000	1.0000	23	0.0115	0.0015
6	0.0000	1.0000	24	0.9750	0.1750
7	0.0000	1.0000	25	80.0000	40.0000
8	0.0000	1.0000	26	0.0500	0.0300
9	0.0000	1.0000	27	0.0500	0.0300
10	0.0800	0.0200	28	0.0700	0.0300
11	0.0105	0.0025	29	0.0500	0.0300
12	0.0700	0.0100	30	55.0000	15.0000
13	0.0105	0.0025	31	1.0000	0.2000
14	2400.0000	400.0000	32	0.4600	0.0600
15	2600.0000	600.0000	33b	0.0000	1.0000
16	0.1250	0.0750	33c	0.0000	1.0000
17	0.1750	0.1250			

$$x_j = \tilde{x}_j * halfrange + midrange, \tag{A.3}$$

$$\tilde{x}_j = \frac{x_j - midrange}{halfrange} \tag{A.4}$$

where $\tilde{x}_j$ represents the scaled variable.

APPENDIX B

Complete Objective Piecewise Linear MARS function

In this appendix, the complete Objective Piecewise Linear MARS function is shown.

Minimize: $0.778713 + 0.183098 * B_1 + 0.508493 * B_2 + 0.082256 * B_3 + 0.098260 * B_4 + 0.290244 * B_5 - 0.087319 * B_6 + 0.184992 * B_7 - 0.043532 * B_8$

where:

$$B_1 = [1 * (-x32 + 1)],$$

$$B_2 = [-1 * (0.75 - 0.5x32 + 0.75x30)],$$

$$B_3 = [1 * (0.75 - 0.5x32 + 0.75x30)],$$

$$B_4 = [-1 * (-0.25 - 0.5x32 + 1.75x24)],$$

$$B_5 = [1 * (-0.25 - 0.5x32 + 1.75x24)],$$

$$B_6 = [1 * (-x14)],$$

$$B_7 = [1 * (x14)],$$

$$B_8 = [1 * (-x9 + 1)]$$

APPENDIX C

Objective Function Graphs

In this appendix, the graphs of the objective function approximated by the Stepwise Linear Regression model and the Piecewise Linear MARS model are presented. Variables 24 and 30 appear in both models.

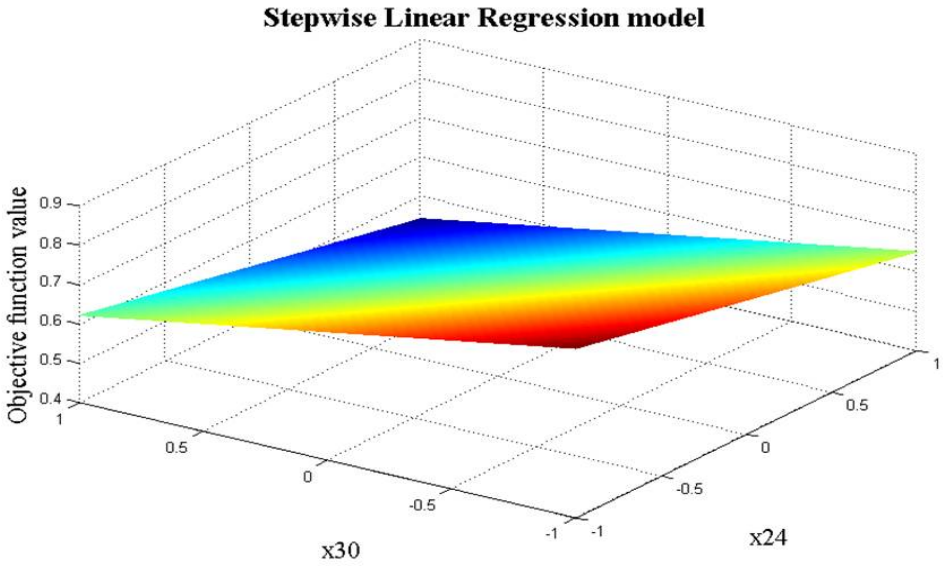

Figure C.1. Stepwise Linear Regression model.

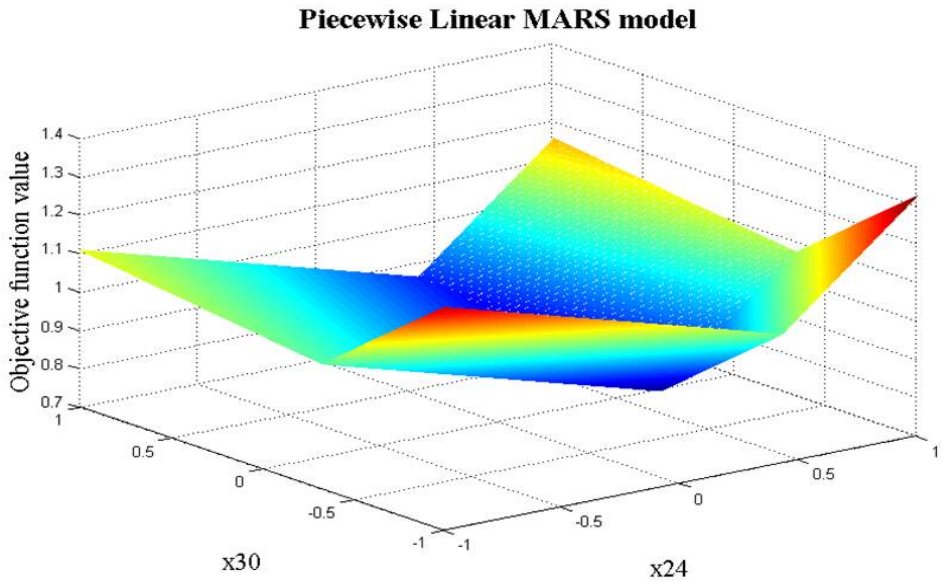

Figure C.2. Piecewise Linear MARS model.

REFERENCES

[1] G. Wang and S. Shan, "Review of metamodeling techniques in support of engineering design optimization," *ASME Transactions, Journal of Mechanical Design*, vol. 129, no. 4, pp. 370–380, 2007.

[2] L. Gu, "A comparison of polynomial based regression models in vehicle safety analysis," *ASME Design Engineering Technical Conferences - Design Automation*, 2001.

[3] 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, no. 1, pp. 38–53, 1999.

[4] C. Cervellera, A. Wen, and V. C. P. Chen, "Neural network and regression spline value function approximations for stochastic dynamic programming," *Computers and Operations Research*, vol. 34, no. 1, pp. 70–90, 2007.

[5] 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*, 2009.

[6] S. Siddappa, D. Gnther, J. Rosenberger, and V. C. P. Chen, "Refined experimental design and regression splines method for network revenue management," *Journal of Pricing and Revenue Management*, vol. 6, no. 3, pp. 188–199, 2007.

[7] V. L. Pilla, J. M. Rosenberger, V. C. P. Chen, and B. C. Smith, "A statistical computer experiments approach to airline fleet assignment," *IIE Transactions*, 2008.

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

[9] T. D. Shih, "Convex versions of multivariate adaptive regression splines and implementation for complex optimization problems," *PhD thesis, Department of Industrial and System Engineering, UT Arlington*, 2006.

[10] A. H. Land and A. G. Doig, "An automatic method of solving discrete programming problems," *Econometrica*, vol. 28, no. 3, pp. 497–520, 1960.

[11] H. E. Romeijn and P. M. Pardalos, *Handbook of Global Optimization*. Dordrecht, The Netherlands: Kluwer Academic Publishers, 1995, vol. 2.

[12] R. Horst, P. M. Pardalos, and N. Thoai, *Introduction to Global Optimization*. Dordrecht, The Netherlands: Kluwer Academic Publishers, 2000.

[13] C. A. Floudas, *Deterministic Global Optimization: Theory, Methods and Applications*. Dordrecht, The Netherlands: Kluwer Academic Publishers, 2000.

[14] C. A. Floudas and P. M. Pardalos, "Recent developments in deterministic global optimization and their relevance to process design," *AICHE Symposium Series*, vol. 96, no. 323, pp. 84–98, 2000.

[15] P. Hansen, B. Jaumard, and S.-H. Lu, "Global optimization of univariate lipschitz functions: I. survey and properties,," *Mathematical Programming*, vol. 55, pp. 251–272, 1992.

[16] K. Hoffman, "A method for globally minimizing concave functions over convex sets," *Mathematical Programming*, vol. 20, pp. 22–32, 1981.

[17] M. A. Duran and I. E. Grossmann, "An outer-approximation algorithm for a class of mixed-integer nonlinear programs," *Mathematical Programming*, vol. 36, pp. 307–339, 1986.

[18] R. Horst, N. Thoai, and H. Tuy, "On an outer-approximation concept in global optimization," *Optimization*, vol. 20, pp. 255–264, 1989.

[19] J. E. Falk and R. M. Soland, "An algorithm for separable nonconvex programming problems," *Management Science*, vol. 15, pp. 550–569, 1969.

[20] G. P. McCormick, "Computability of global solutions to factorable nonconvex programs: Part i-convex underestimating problems," *Mathematical Programming*, vol. 10, pp. 147–175, 1976.

[21] F. A. Al-Khayyal and J. E. Falk, "Jointly constrained biconvex programming," *Mathematics of Operations Research*, vol. 8, 1983.

[22] O. K. Gupta and A. Ravindran, "Branch and bound experiments in convex nonlinear integer programming," *Management Science*, vol. 31, pp. 1533–1546, 1985.

[23] R. Horst and H. Tuy, "On the convergence of global methods in multiextremal optimization," *Journal of Optimization Theory and Applications*, vol. 54, no. 253, 1987.

[24] F. A. Al-Khayyal, "Jointly constrained bilinear programs and related problems: An overview," *Computers in Mathematical Applications*, vol. 19, no. 53, 1990.

[25] E. R. Hansen, "Global optimization using interval analysis," *Pure and Applied Mathematics*, 1992.

[26] H. S. Ryoo and N. V. Sahinidis, "A branch-and-reduce approach to global optimization," *Journal of Global Optimization*, vol. 8, pp. 107–139, 1996.

[27] R. B. Kearfott, *Rigorous Global Search: Continuous Problems*. Dordrecht, The Netherlands: Kluwer Academic Publishers, 1996.

[28] T. G. W. Epperly and R. E. Swaney, *Branch and bound for global NLP: New bounding LP*. Dordrecht, The Netherlands: Kluwer Academic Publishers, 1996.

[29] C. S. Adjiman, S. Dallwig, C. A. Floudas, and A. Neumaier, "A global optimization method, αbb, for general twice-differentiable constrained nlps-i. theoretical advances," *Computers and Chemical Engineering*, vol. 22, pp. 1137–1158, 1998.

[30] M. Tawarmalani and N. V. Sahinidis, "Global optimization of mixed integer nonlinear programs: A theoretical and computational study," *Mathematical Programming*, 1999.

[31] H. Tuy, "Concave programming under linear constraints," *Doklady Akademic Nauk*, vol. 159, pp. 32–35, 1964.

[32] R. J. Hillestad and S. E. Jacobsen, "Reverse convex programming," *Applied Mathematics and Optimization*, vol. 6, pp. 63–78, 1980.

[33] H. Tuy, T. V. Thieu, and N. Q. Thai, "A conical algorithm for globally minimizing a concave function over a closed convex set," *Mathematics of Operations Research*, vol. 10, pp. 498–514, 1985.

[34] H. Tuy, "Global optimization of a difference of two convex functions," *Mathematical Programming Study*, vol. 30, pp. 150–182, 1987.

[35] V. Visweswaran and C. A. Floudas, "New properties and computational improvement of the gop algorithm for problems with quadratic objective functions and constraints," *Journal of Global Optimization*, vol. 3, pp. 439–462, 1993.

[36] H. D. Sherali and A. Alameddine, "A new reformulation-linearization technique for bilinear programming problems," *Journal of Global Optimization*, vol. 2, pp. 379–410, 1992.

[37] H. D. Sherali and C. H. Tuncbilek, "A global optimization algorithm for polynomial programming problems using a reformulation-linearization technique," *Journal of Global Optimization*, vol. 2, pp. 101–112, 1992.

[38] E. R. Hansen, "Global optimization using interval analysis: The one-dimensional case," *Journal of Optimization Theory and Applications*, vol. 29, no. 331, 1979.

[39] R. Horst and H. Tuy, *Global Optimization - Deterministic Approaches*. Berlin / Heidelberg / New York, 3rd edition: Springer, 1996.

[40] Q. Zheng and D. Zhuang, "Integral global minimization: algorithms, implementations and numerical tests," *Journal of Global Optimization*, vol. 7, pp. 421–454, 1995.

[41] N. Z. Shor, "Dual quadratic estimates in polynomial and boolean programming," *Annals of Operations Research, Special Volume on Computational Methods in Global Optimization, (Eds. P.M. Pardalos and J.B. Rosen)*, vol. 27, 1990.

[42] C. A. Floudas and V. A. Visweswaran, "A global optimization algorithm (gop) for certain classes of nonconvex nlps: I. theory, computers and chemical engineering," vol. 14, no. 12, pp. 1397–1417, 1990.

[43] A. Ben-Tal, G. Eiger, and V. Gershovitz, "Global minimization by reducing the duality gap," *Mathematical Programming*, vol. 63, pp. 193–212, 1994.

[44] D. R. Jones, C. Perttunen, and B. Stuckman, "Lipschitzian optimization without the lipschitz constant," *Journal of Global Optimization Theory and Applications*, vol. 79, pp. 157–181, 1993.

[45] C. Guus, E. Boender, and H. E. Romeijn, *Stochastic methods in R. Horst and P. M. Pardalos, editors*. Dordrecht, The Netherlands: Kluwer Academic Publishers, 1995.

[46] S. Brooks, "A discussion of random methods for seeking maxima," *Operations Research*, vol. 6, pp. 244–251, 1958.

[47] J. Matyas, "Random optimization," *Automation Remote Control*, vol. 26, pp. 246–253, 1965.

[48] L. A. Rastrigin and Y. Rubinstein, "The comparison of random search and stochastic approximation while solving the problem of optimization," *Automatic Control*, vol. 2, pp. 23–29, 1969.

[49] Z. B. Zabinsky, "Stochastic methods for practical global optimization," *Journal of Global Optimization*, vol. 13, pp. 433–444, 1998.

[50] A. H. G. R. Kan and G. T. Timmer, "Stochastic global optimization methods. i: Clustering methods," *Mathematical Programming*, vol. 39, pp. 27–56, 1987.

[51] ——, "Stochastic global optimization methods. i: Multi level methods," *Mathematical Programming*, vol. 39, pp. 57–78, 1987.

[52] H.-L. Li and C.-T. Chou, "A global approach for nonlinear mixed discrete programming in design optimization," *Engineering Optimization*, vol. 22, pp. 109–122, 1994.

[53] K. A. Dill, A. T. Phillips, and J. B. Rosen, *Molecular structure prediction by global optimization.* Dordrecht, The Netherlands: Kluwer Academic Publishers, 1997.

[54] J. J. More and Z. Wu, "Global continuation for discrete geometry problems," *SIAM Journal on Optimization*, vol. 7, pp. 814–836, 1997.

[55] A. A. Zhigljavsky, *Theory of Global Random Search.* Dordrecht, The Netherlands: Kluwer Academic Publishers, 1991.

[56] J. D. Pinter, *Global Optimization in Action (Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications).* Dordrecht, The Netherlands: Kluwer Academic Publishers, 1996.

[57] A. V. Levy and S. Gomez, "The tunneling method applied for the global minimization of functions. in boggs, p.t., editor," *Numerical Optimization.*

[58] E. Rich and K. Knight, *Artificial Intelligence.* McGraw-Hill, 1991.

[59] S. Kirkpatrick, C. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.

[60] F. Glover, "Heuristics for integer programming using surrogate constraints," *Decision Sciences*, vol. 8, pp. 156–166, 1977.

[61] J. H. Holland, "Adaptation in natural and artiffcial systems," 1975.

[62] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison Wesley Publication Co., Reading, M.A., 1989.

[63] S.-J. Wu and P.-T. Chow, "Genetic algorithms for nonlinear mixed discrete-integer optimization problems via meta-genetic parameter optimization," *Engineering Optimization*, vol. 24, pp. 137–159, 1995.

[64] B. K.-S. Cheung, A. Langevin, and H. Delmaire, "Coupling genetic algorithm with a grid search method to solve mixed integer nonlinear programming problems," *Comput. Math. Appl*, vol. 34, pp. 13–23, 1997.

[65] I. Rechenberg, "Evolutionsstrategie: Optimierung technischer systeme nach prinzipien der biologischen evolution," *Frommann-Holzboog*, 1973.

[66] M. Dorigo, "Ottimizzazione, apprendimento automatico, ed algoritmi basati su metafora naturale," *PhD thesis, Politecnico di Milano, Italy*, 1992.

[67] J. Kennedy and R. Eberhart, "Particle swarm optimization," *In IEEE International Conference on Neural Networks*, pp. 1942–1948, 1995.

[68] R. Storn and K. Price, "Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341–359, 1997.

[69] A. Guinta, "Aircraft multidisciplinary design optimization using design of experiments theory and response surface modeling," *Dissertation submitted to the faculty of Virginia Polytechnic Institute and State University*, 1997.

[70] A. Giunta, L. T. Watson, and J. Koehler, "A comparison of approximation modeling techniques: Polynomial versus interpolating models," *7th AIAA/USAF/NASA/ISSMO Symposium on Multid isciplinary Analysis and Optimization*, vol. 1, pp. 392–404, 1998.

[71] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global Optimization*, vol. 13, pp. 455–492, 1998.

[72] R. Jin, W. Chen, and T. Simpson, "Comparative studies of meta-modeling techniques under multiple modeling criteria," *Journal of Structural and Multidisciplinary Optimization*, vol. 23, no. 1, pp. 1–13, 2001.

[73] S. Hosder, L. T. Watson, B. Grossman, W. H. Mason, H. Kim, R. T. Haftka, and S. E. Cox, *Polynomial Response Surface Approximations for the Multidisciplinary Design Optimization of a High Speed Civil Transport.* Dordrecht, The Netherlands: Kluwer Academic Publishers, 2001, vol. 2.

[74] D. R. Jones, "A taxonomy of global optimization methods based on response surfaces," *Journal of Global Optimization*, vol. 21, pp. 345–383, 2001.

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

[76] M. Bjorkman and K. Holmstrom, "Global optimization of costly non-convex functions using radial basis functions," *Journal of Global Optimization*, vol. 19, pp. 201–227, 2001.

[77] M. Emmerich, A. Giotis, M. Ozdemir, T. Back, and K. Giannakoglou, "Metamodel assisted evolution strategies," *Parallel Problem Solving from Nature V (PPSN VII), Springer-Verlag*, pp. 362–370, 2002.

[78] L. Willmes, T. Back, Y. Jin, and B. Sendhoff, "Comparing neural networks and kriging for fitness approximation in evolutionary optimization," *IEEE Congress on Evolutionary Computation*, pp. 663–670, 2003.

[79] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. K. Tucker, "Surrogate-based analysis and optimization," *Progress in Aerospace Sciences*, vol. 41, pp. 1–28, 2005.

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

[81] S. Crino and D. E. Brown, "Global optimization with multivariate adaptive regression splines," *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, vol. 37, no. 2, 2007.

[82] A. I. Forrester and A. J. Keane, "Recent advances in surrogate-based optimization," *Progress in Aerospace Sciences*, vol. 45, pp. 50–79, 2009.

[83] M. Holena, D. Linke, U. Rodemerck, and L. Bajer, *Neural Networks as Surrogate Models for Measurements in Optimization Algorithms*, ser. Analytical and Stochastic Modeling Techniques and Applications, 2010.

[84] M. A. Abramson, T. J. Asaki, J. E. D. Jr., R. M. Jr., and M. J. Sottile, "An efficient class of direct search surrogate methods for solving expensive optimization problems with cpu-time-related functions," *Struct Multidisc Optim Research Paper*, 2011.

[85] J. Gu, G. Y. Li, and Z. Dong, "Hybrid and adaptive meta-model-based global optimization," *Engineering Optimization*, vol. 44, no. 1, pp. 87–104, 2012.

[86] M. L. Bergamini, I. Grossmann, N. Scenna, and P. Aguirre, "An improved piecewise outer-approximation algorithm for the global optimization of minlp models involving concave and bilinear terms," *Computers and Chemical Engineering*, vol. 32, pp. 477–493, 2008.

[87] A. Martin, M. Moller, and S. Moritz, "Mixed integer models for the stationary case of gas network optimization," *Math. Programming*, vol. 105, pp. 563–582, 2006.

[88] K. L. Croxton, B. Gendron, and T. L. Magnanti, "Models and methods for merge-in-transit operations," *Transportation Science*, vol. 37, no. 1, 2003.

[89] A. B. Keha, I. R. de Farias, and G. L. Nemhauser, "A branch-and-cut algorithm without binary variables for nonconvex piecewise linear optimization," *Operations Research*, vol. 54, pp. 847–858, 2006.

[90] K. L. Croxton, B. Gendron, and T. L. Magnanti, "A comparison of mixed-integer programming models for nonconvex piecewise linear cost minimization problems," *Management Sci.*, vol. 49, pp. 1268–1273, 2003.

[91] A. B. Keha, I. R. de Farias, and G. L. Nemhauser, "Models for representing piecewise linear cost functions," *Operations Research Letters*, vol. 32, pp. 44–48, 2004.

[92] J. Vielma, A. Keha, and G. Nemhauser, "Nonconvex, lower semi-continuous piecewise linear optimization," *Discrete Optimization*, vol. 5, pp. 467–488, 2008.

[93] D. A. Babayev and G. I. Bell, "An optimization problem with a separable nonconvex objective function and a linear constraint," *Journal of Heuristics*, vol. 7, pp. 169–184, 2001.

[94] J. P. Vielma, S. Ahmed, and G. L. Nemhauser, "Mixed-integer models for nonseparable piecewise-linear optimization: Unifying framework and extensions," *Operations Research*, vol. 58, no. 2, pp. 303–315, 2010.

[95] H. D. Sherali and H. Wang, "Global optimization of non-convex factorable programming problems," *Math. Program., Ser. A*, vol. 89, pp. 459–478, 2001.

[96] H. D. Sherali and V. Ganesan, "A pseudo-global optimization approach with application to the design of containerships," *Journal of Global Optimization*, vol. 26, pp. 335–360, 2003.

[97] H. Zhang and S. Wang, "Linearly constrained global optimization via piecewise-linear approximation," *Journal of Computational and Applied Mathematics*, vol. 214, pp. 111–120, 2008.

[98] R. M. Chrysanthos, E. Gounaris, and C. Floudas, "Global optimization of gas lifting operations: A comparative study of piecewise linear formulations," *Ind. Eng. Chem. Res*, vol. 48, pp. 6098–6104, 2009.

[99] L. Gu, R. Yang, C. Tho, M. Makowskit, O. Faruquet, and Y.Li, "Optimization and robustness for crashworthiness of side impact," *International Journal of Vehicle Design*, vol. 26, no. 4, pp. 348–360, 2001.

[100] R. J. Yang, N. Wang, C. H. Tho, J. P. Bobineau, and B. P. Wang, "Metamodeling development for vehicle frontal impact simulation," *Journal of Mechanical Design*, vol. 127, no. 5, pp. 1014–1020, 2005.

[101] X. Liao, L. Qing, Y. Xujing, Z. Weigang, and L. Wei, "Multiobjective optimization for crash safety design of vehicles using stepwise regression model," *Structural and Multidisciplinary Optimization*, vol. 35, no. 6, pp. 561–569, 2008.

[102] D. Martinez, *MARS Variants: Convex vs. Non-Convex and Piecewise-Linear vs. Smooth*, Charlotte, NC., 2011.

[103] T. D. Shih, D. Martinez, and V. C. P. Chen, "A convex version of multivariate adaptive regression splines," *COSMOS Technical Report 12-01 (Submitted)*, 2012.

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

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

[106] R. J. Yang, L. Gu, L. Liaw, C. Gearhart, C. H. Tho, X. Liu, and B. P. Wang, "Approximations for safety optimization of large systems," *ASME 2000 Design Engineering Technical Conferences - Design Automation Conference*, 2000.

[107] K. Hamza and K. Saitou, "Crashworthiness design using meta-models for approximating the response of structural members," *University Conference on Mechanical Design and Production*, 2004.

[108] ——, "Vehicle crashworthiness design via a surrogate model ensemble and a co-evolutionary genetic algorithm," *Proceedings of the ASME Design Engineering Technical Conferences*, 2005.

[109] J. J. D. Aspenberg and L. Nilsson, "Robust optimization of front members in a full frontal car impact," *Engineering Optimization*, vol. 45, pp. 245–264, 2013.

[110] G. Syswerda, *Uniform crossover in genetic algorithms*, 1989.

[111] F. Rothlauf, *Representations for Genetic and Evolutionary Algorithms.* Second edition: Springer, 1996.

[112] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs.* Third, revised and extended edition: Springer, 1996.

[113] J. J. Grefenstette, "Optimization of control parameters for genetic algorithms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 16, pp. 122–128, 1986.

BIOGRAPHICAL STATEMENT

Nadia Martinez was born in Mexico City and raised in Saltillo Coahuila, where she earned her B.S. degree in Industrial Engineering at the Instituto Tecnolgico de Saltillo. She came to United States on January 2007 to enrich her education at The University of Texas at Arlington (UTA). She received her M.S. degree in fall 2008 and she is currently working on her Ph.D. at the Industrial and Manufacturing Systems Engineering Department. She has also worked as a Graduate Research Assistant (GRA) at TMAC, which is a research center of the College of Engineering at UTA, where she has participated in different projects related to her Industrial Engineering career.