MARS KNOT POSITIONING AND GLOBAL OPTIMIZATION

by

XINGLONG JU

DISSERTATION

Submitted in partial fulfillment of the requirements

for the degree of Doctor of Philosophy at

The University of Texas at Arlington

August, 2019

Arlington, Texas

Supervising Committee:

Victoria Chen, Supervising Professor

Jay Rosenberger, Supervising Professor

Kamesh Subbarao

Chen Kan

Copyright by

Xinglong Ju

2019

ACKNOWLEDGEMENTS

This research is partially funded by National Science Foundation grant CMMI-1434401. I thank my academic advisor, Dr. Sheik Imrhan for five years' advising and help. I thank my Supervising Committee members, Dr. Kamesh Subbarao and Dr. Chen Kan for the advice on the dissertation. I thank my Supervising Professors, Dr. Victoria Chen and Dr. Jay Rosenberger for the help in the study, the research and the life. I thank Dr. Feng Liu for the help in the research. I thank Dr. Bonnie Boardman and Dr. Shouyi Wang for the help in the study.

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

MARS KNOT POSITIONING AND GLOBAL OPTIMIZATION

Xinglong Ju, Ph.D.

The University of Texas at Arlington, 2019

Supervising Professors: Victoria Chen, Jay Rosenberger

Multivariate adaptive regression splines (MARS) is a statistical modeling approach with wide real-world applications. In the MARS model building process, knot positioning is a critical step that potentially affects the accuracy of the final MARS model. Identifying well-positioned knots entails assessing the quality of many knots in each model building iteration, which requires much computation efforts. By exploring the change in the residual sum of squares (RSS) within MARS, we find that local optima from previous iterations can be very close to those of the current iteration. In our approach, the prior change in RSS information is used to "warm start" an optimal knot positioning. We propose two methods for MARS knot positioning. The first method is a hill climbing method (HCM), which ignores prior change in RSS information. The second method is a hill climbing method using prior change in RSS information (PHCM). Numerical experiments are conducted on data with up to 30 dimensions. Our results show that both versions of hill climbing methods outperform Chen's MARS knot selection method on datasets with different noise levels. Further, PHCM using prior change in RSS information performs best in both accuracy and computational speed.

Multivariate adaptive regression splines (MARS) is a flexible statistical

modeling method that has been popular for data mining applications. MARS has also been employed to approximate unknown relationships in optimization for complex systems, including surrogate optimization, dynamic programming, and two-stage stochastic programming. Given the increasing desire to optimize real world systems, this paper presents an approach to globally optimize a MARS model that allows up to two-way interaction terms that are products of truncated linear univariate functions (TITL-MARS). Specifically, such a MARS model consists of linear and quadratic structure. This structure is exploited to formulate a mixed integer quadratic programming problem (TITL-MARS-OPT). To appreciate the contribution of TITL-MARS-OPT, one must recognize that popular heuristic optimization approaches, such as evolutionary algorithms, do not guarantee global optimality and can be computationally slow. The use of MARS maintains the flexibility of modeling within TITL-MARS-OPT while also taking advantage of the linear modeling structure of MARS to enable global optimality. Computational results compare TITL-MARS-OPT with a genetic algorithm for two types of cases. First, a wind farm power distribution case study is described and then other TITL-MARS forms are tested. The results show the superiority of TITL-MARS-OPT over the genetic algorithm in both accuracy and computational time.

TABLE OF CONTENTS

CHAPTER 1

Introduction

1.1 MARS knot positioning

Various regression methods are used to build statistical models of real world problems to study relationships between some potential factors and corresponding results, which include simple linear regression method (SLR), multiple linear regression method (MLR), logistic regression method, deep neural network, etc. Multivariate adaptive regression splines (MARS) [1] is one of the most popular regression methods which performs very well when dealing with high dimensional data and it has been used within many applications, such as predicting mobile radio channels [2], predicting distributions of freshwater diadromous fish [3], analyzing relationships between the distributions of 15 freshwater fish species and their environment [4], mining the customer credit [5], modeling direct response behavior [6], etc.

The MARS regression process is iterative where usually in each iteration a

pair of basis function bending over a knot will be added to the MARS model. But if the knot is at the margin, only one basis function will be added to the model because the other basis function is simply 0.

Knot positioning is a time-consuming component in MARS, especially for estimating more complex forms, and highly affects the accuracy of the resulting MARS model. In this research, it is desired to reduce the computational effort of knot positioning, while maintaining an accurate model [7]. Friedman [1] used all data values in the input data as candidate knot locations and used a greedy algorithm to select specific knot positions among the candidates. The knot position that provided the greatest improvement in the residual sum of squares was selected in each iteration of the MARS algorithm. Selecting a knot position from all possible data values is time-consuming when the data set is large. Chen et al. [8] used a fixed number of candidate knots that was a subset of the data values, such that the candidate knots were equally spaced. It is also possible to select a subset of the data values, such that candidate knots have some minimum number of data values between them (referred to as MinSpan in the R code "earth"). This speeds up the knot positioning process, but may miss some potentially superior knot positions that could achieve a more accurate MARS model. Koc and Iyigun [9] introduced a mapping approach to use more representative data points as candidate knots in the MARS knot positioning process. This approach can yield efficiency in knot positioning when the data are not evenly scattered over the input space. Miyata and Shen [10] proposed knot optimization using an evolutionary algorithm. Their approach can be generally applied for various forms of spline basis functions,

but was only demonstrated for one input dimension and required additional computational effort over existing approaches. Of interest for the current work was the positioning of knots *between* data values, which greatly expanded the knot search space [10].

We propose two new methods for MARS knot positioning that seek to reduce the computational effort of knot positioning without degrading the quality of fit. We refer to these methods as the hill climbing method (HCM) and the climbing method using prior information (PHCM) where the objective is to decrease the residual sum of squares (RSS). Numerical experiments using different dataset sizes and different numbers of candidate knots with different noise levels are investigated in this paper.

1.2 MARS global optimization

Global optimization is to find the best solution for a given objective, bounds and criteria where different global optimization methods are applied to different problems. For a network problem, Dijkstra's algorithm can be used to find shortest path. Simplex, dual simplex and big-M methods are used to get a global optimum for the assignment problem. Gradient descent methods are used to solve convex problems.

It is also of paramount importance to find an analytical way to optimize on the achieved MARS model when making decisions based on the model. Martinez et al. [11] proposed a global optimization method using mixed integer programming for piece-wise linear MARS, which is used in a car crash

problem as an application. However, this method could not be applied to two-way interaction truncated linear MARS (TITL-MARS) optimization problem because TITL-MARS is a quadratic problem.

We propose a novel mixed integer quadratic programming method to solve the global optimization problem of two-way interaction MARS with truncated linear basis function (TITL-MARS-OPT). We compare the proposed optimization method with genetic algorithm under different wind farm power distribution functions and other mathematical functions and the results show TITL-MARS-OPT method works better than genetic algorithm in both accuracy and time efficiency.

Bibliography

[1] Jerome H Friedman. Multivariate adaptive regression splines. *The annals of statistics*, pages 1–67, 1991.

[2] Torbjörn Ekman and Gernot Kubin. Nonlinear prediction of mobile radio channels: measurements and mars model designs. In *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, volume 5, pages 2667–2670. IEEE, 1999.

[3] JR Leathwick, D Rowe, J Richardson, Jane Elith, and T Hastie. Using multivariate adaptive regression splines to predict the distributions of new zealand's freshwater diadromous fish. *Freshwater Biology*, 50(12):2034–2052, 2005.

[4] JR Leathwick, J Elith, and T Hastie. Comparative performance of generalized additive models and multivariate adaptive regression splines for statistical modelling of species distributions. *Ecological modelling*, 199(2): 188–196, 2006.

[5] Tian-Shyug Lee, Chih-Chou Chiu, Yu-Chao Chou, and Chi-Jie Lu. Mining the customer credit using classification and regression tree and multivariate adaptive regression splines. *Computational Statistics & Data Analysis*, 50(4):1113–1130, 2006.

[6] Joel Deichmann, Abdolreza Eshghi, Dominique Haughton, Selin Sayek, and Nicholas Teebagy. Application of multiple adaptive regression splines (mars) in direct response modeling. *Journal of Interactive Marketing*, 16 (4):15–27, 2002.

[7] Xinglong Ju, Victoria CP Chen, Jay M Rosenberger, and Feng Liu. Knot optimization for multivariate adaptive regression splines. In *IISE Annual Conference. Proceedings*. Institute of Industrial and Systems Engineers (IISE), 2019.

[8] Victoria CP Chen, David Ruppert, and Christine A Shoemaker. Applying experimental design and regression splines to high-dimensional continuous-state stochastic dynamic programming. *Operations Research*, 47(1):38–53, 1999.

[9] Elcin Kartal Koc and Cem Iyigun. Restructuring forward step of mars

algorithm using a new knot selection procedure based on a mapping approach. *Journal of Global Optimization*, 60(1):79–102, 2014.

[10] Satoshi Miyata and Xiaotong Shen. Free-knot splines and adaptive knot selection. *Journal of the Japan Statistical Society*, 35(2):303–324, 2005.

[11] Nadia Martinez, Hadis Anahideh, Jay M Rosenberger, Diana Martinez, Victoria CP Chen, and Bo Ping Wang. Global optimization of non-convex piecewise linear regression splines. *Journal of Global Optimization*, 68(3): 563–586, 2017.

CHAPTER 2

FAST KNOT OPTIMIZATION FOR MULTIVARIATE ADAPTIVE REGRESSION SPLINES USING HILL CLIMBING METHODS

Xinglong Ju, Victoria C. P. Chen, Jay M. Rosenberger, Feng Liu

2.1 Abstract

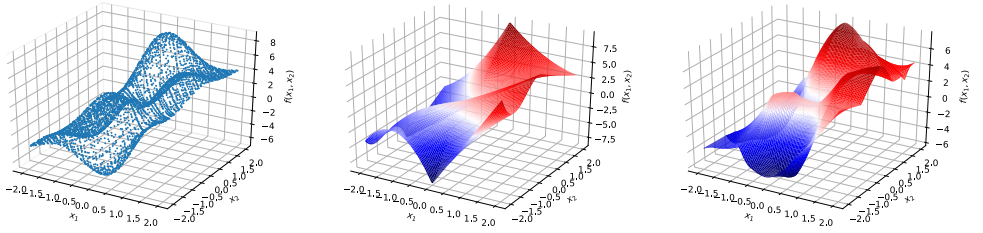
Multivariate adaptive regression splines (MARS) is a statistical modeling approach with wide real-world applications. In the MARS model building process, knot positioning is a critical step that potentially affects the accuracy of the final MARS model. Identifying well-positioned knots entails assessing the quality of many knots in each model building iteration, which requires much computation efforts. By exploring the change in the residual sum of squares (RSS) within MARS, we find that local optima from previous iterations can be very close to those of the current iteration. In our approach, the prior change in RSS information is used to "warm start" an optimal knot positioning. We propose two methods for MARS knot positioning. The first method is a hill climbing method (HCM), which ignores prior change in RSS information. The second method is a hill climbing method using prior change in RSS information (PHCM). Numerical experiments are conducted on data with up to 30 dimensions. Our results show that both versions of hill climbing methods outperform Chen's MARS knot selection method on datasets with different noise levels. Further, PHCM using prior change in RSS information performs best in both accuracy and computational speed.

2.2 Introduction

As a popular non-parametric regression technique, multivariate adaptive regression splines (MARS) algorithm was first introduced by Friedman in 1991 [1]. Because of its flexibility and accuracy, MARS has been used in many studies

including predicting distributions of freshwater diadromous fish [2], analyzing relationships between the distributions of 15 freshwater fish species and their environment [3], mining the customer credit [4], modeling direct response behavior [5], building a decision-making framework for ozone pollution control [6], assessment of gully erosion susceptibility [7], estimating heating load in buildings [8], modeling daily dissolved oxygen concentration [9] etc.

Knot positioning is a time-consuming step in the MARS building processing, and it highly affects the accuracy of the final MARS model. The situation can getting worse for high dimensional regression model. In this research, it is desirable to reduce the computational cost of knot positioning, while maintaining an accurate model. Friedman [1] used all values from the predictive variables as candidate knot locations and used a greedy algorithm to select specific knot positions among all the candidates. The knot position that provides the greatest improvement in the residual sum of squares (RSS) was selected in each iteration of the MARS algorithm. Selecting a knot position from all possible data values is time-consuming when the data set is large. Chen et al. [10] used a fixed number of candidate knots that was a subset of the data values, such that the candidate knots were equally spaced. It is also possible to select a subset of the data values, such that candidate knots have the minimum number of data values between them (referred to as MinSpan in the R code "earth"), which can speed up the knot positioning process, but may miss some potentially superior knot positions that could achieve a more accurate MARS model. Koc and Iyigun [11] introduced a mapping approach to use more representative data points as candidate knots in the MARS knot



(a) Training dataset (b) Model 1: R^2=0.94 (c) Model 2: R^2=0.99

Figure 2.1: MARS model with different knot positions

positioning process. This approach can yield efficiency in knot positioning when the data are not evenly scattered over the input space. Miyata and Shen [12] proposed knot optimization using an evolutionary algorithm. Their approach can be generally applied for various forms of spline basis functions, but was only demonstrated for one input dimension and required additional computational effort compared to existing approaches.

If knots are not positioned well, the MARS model may not represent the relationships properly because basis functions will only bend at these positions. Suppose we fit two input dimensions, as illustrated in Figure 2.1a, where x_1 and x_2 are the input variables, and the surface has multiple peaks and valleys. MARS models with different knot positions are shown in Figures 2.1b and 2.1c. MARS model 1 achieved a coefficient of determination of $R^2 = 0.94$, and MARS model 2 achieved $R^2 = 0.99$, which indicates that MARS model 2 is better fit to the data than MARS model 1, as can be seen visually in the figures. Hence, limiting the set of candidate knots can degrade the model fit; however, an exhaustive search of knot positions is computational expensive.

In this research, we propose improved knot positioning mechanisms during the MARS building process. We propose two new methods for MARS

knot positioning that seek to reduce the computational effort of knot positioning without degrading the quality of fit. We refer to these methods as the hill climbing method (HCM) and the hill climbing with prior information (PHCM) where the objective is to decrease the RSS. Numerical experiments using different dataset sizes and different numbers of candidate knots with different noise levels are investigated in this paper.

The rest of the paper are organized as follows. In Section 2.3, the original MARS algorithm is introduced. Section 2.4 provides the description of the datasets. In Section 2.5, the knot optimization for MARS using hill climbing methods is described in detail. Section 2.6 presents the experimental results, and, finally, concluding remarks are given in Section 2.7.

2.3 MARS background

MARS is introduced for the regression setting with multiple input variables and a response variable. In MARS model, the approximated MARS function is composed from a linear model of basis functions, which is defined from hinge functions or multiplication of hinge functions. The MARS model can be written as follows:

$$\hat{f}(\mathbf{x}) = \sum_{m=0}^{M} \{a_m \cdot B_m(\mathbf{x})\}, \tag{2.1}$$

where $\hat{f}(\mathbf{x})$ is the MARS model and $B_m(\mathbf{x})$ is called the basis function. Here m denotes the index of the basis function and M indicates the total number of basis functions in the MARS model. The coefficient of m-th basis function is denoted as a_m and $\mathbf{x} \in \mathbb{R}^n$ denotes the predicting variable vector. MARS

uses a product form for the basis function:

$$B_m(\mathbf{x}) = \prod_{k=1}^{K_m} b_{k,m}. \tag{2.2}$$

Here $b_{k,m}$ is the k-th univariate function in $B_m(\mathbf{x})$ and K_m denotes the total number of univariate functions in $B_m(\mathbf{x})$. When $K_m = 1$, then the basis function is univariate. Otherwise, K_m is the degree of the interaction term.

In each basis function, the refraction points are the knots for the basis function. The $b_{k,m}$ are truncated linear functions of the form:

$$b(x|t) = [+(x-t)]_+ = \max\{+(x-t), 0\}, \tag{2.3}$$

or

$$b(x|t) = [-(x-t)]_+ = \max\{-(x-t), 0\}, \tag{2.4}$$

where the location t is called **knot** for the basis function.

Let $\{\mathbf{x}_i, y_i\}_{i=1}^N$ represent a dataset, where $\mathbf{x}_i \in \mathbb{R}^n$ denotes the i-th data point in predicting variable dataset, and the i-th data point for the response variable is defined as y_i. The sample size is denoted as N and i is the index of the data point ($i = 1, 2, 3, \ldots, N$).

The residual sum of squares between the observed value and the predicted value, denoted as e, is defined as:

$$e = \frac{1}{N} \sum_{i=1}^{N} \left[y_i - \hat{f}(\mathbf{x}_i) \right]^2. \tag{2.5}$$

In general, a smaller e is considered to be a better fit to the data. In MARS, a

penalty term with e is used to avoid overfitting, but e alone is used for selecting among knot positions within the MARS algorithm.

The MARS forward stepwise algorithm [1] using the truncated linear univariate basis function is given in Algorithm 1 where $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$ is the input dataset and $M_{\max}$ is the maximum number of basis functions. In each MARS iteration, the algorithm seeks a pair of basis functions to add to its current set. Candidate basis functions can be new univariate terms or interaction terms that are split from the current set. The innermost loop of the algorithm (line 5-11) considers all possible knot positions for a univariate term or additional split of an interaction term. In line 1 of Algorithm 1, the MARS model starts with a constant. The current best residual sum of squares e* is initialized to be ∞. From line 2 to line 17, it adds basis functions until M_{max} basis functions are added to the MARS model. From line 3 to line 13, the regression process tries to split on all already added basis functions. The set $\{v(k,m)\}_{k=1}^{K_m}$ is the variable index set of the basis function $B_m(\mathbf{x})$ and v denotes the variable index. For example, if

$$B_m(\mathbf{x}) = [+(x_1 - 0.2)]_+ \cdot [-(x_3 - 0.6)]_+, \tag{2.6}$$

then the set

$$\{v(k,m)\}_{k=1}^{K_m} = \{1, 3\}. \tag{2.7}$$

The candidate knot set of the basis function $B_m(\mathbf{x})$ at v-th variable is denoted as $\{\mathbf{x}_{j,v} | B_m(\mathbf{x}_j) > 0\}_{j=1}^{N}$ and it consists of the v-th variable values of the data points which make the basis function positive. In line 7, the new e for MARS

model with new basis functions is calculated . From line 8 to 10, e is compared with e^*. If e is less than $e*$, it indicates the new model is better, and we store the related information, $e*$, the index of the basis function m^*, the variable index v^* and the knot value t^*. In line 14 and line 15, two new basis functions are added to the MARS model.

Algorithm 1: MARS forward stepwise algorithm

Input: $\{\mathbf{x}_i, y_i\}_{i=1}^N$,$M_{max}$
Result: MARS regression model $\hat{f}(\mathbf{x})$

```
1  B_1(x) = 1, M = 1, e* = ∞
2  while M < M_max do
3      for m = 1 to M do
4          for v ∉ {v(k,m)}_{k=1}^{K_m} do
5              for t ∈ {x_{j,v} | B_m(x_j) > 0}_{j=1}^N do
6                  f̂ = Σ_{i=1}^M a_i B_i(x) + a_{M+1} B_m(x)[+(x_v − t)]_+ +
                     a_{M+2} B_m(x)[−(x_v − t)]_+
7                  e = min_{a_1,...,a_{M+2}} e(f̂)
8                  if e < e* then
9                      e* = e, m* = m, v* = v, t* = t
10                 end
11             end
12         end
13     end
14     B_{M+1}(x) = B_{m*}(x)[+(x_{v*} − t*)]_+
15     B_{M+2}(x) = B_{m*}(x)[−(x_{v*} − t*)]_+
16     M = M + 2
17 end
```

2.4 Datasets

In this paper, 7 datasets are used to investigate and verify the proposed new methods. The first 6 datasets are generated from 6 functions and the Sobol sequence is used to sample values in the input space [13]. The seventh dataset

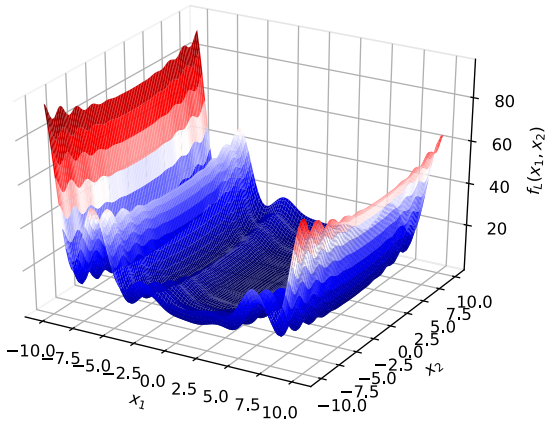


Figure 2.2: $f_L(x_1, x_2)$ $d = 2$

is a wind farm power distribution dataset [14].

The first dataset D_L is generated from the Levy function $f_L(\mathbf{x})$ [15] as

$$\begin{aligned} f_L(\mathbf{x}) =& \sin^2(\pi w_1) + \sum_{i=1}^{d-1} (w_i - 1)^2 \left[1 + 10\sin^2(\pi w_i + 1)\right] + (w_d - 1)^2 \left[\, 1 + \right. \\ & \left. \sin^2(2\pi w_d) \,\right], w_i = 1 + \frac{x_i - 1}{4}, \text{ for all } i = 1, \ldots, d \\ & -10 \leqslant x_i \leqslant 10, \end{aligned} \tag{2.8}$$

where $\mathbf{x}$ is the independent variable. The dimension of $\mathbf{x}$ is denoted as d, and in this paper, $d = 30$ which indicates $\mathbf{x}$ is 30-dimensional. Figure 2.2 is the surface of Levy function when $d = 2$.

Datasets D_1, D_2, D_3, D_4 and D_5 are generated from functions f_1, f_2, f_3, f_4 and f_5 [11], respectively. In dataset D_1, $\mathbf{x}$ has 7 dimensions. In dataset D_2, $\mathbf{x}$ is 10-dimensional. For D_3, $\mathbf{x}$ has 10 dimensions and for D_4, $\mathbf{x}$ is 3-dimensional. For D_5, $\mathbf{x}$ is 21-dimensional with $\boldsymbol{\alpha} = \{0.15, -0.96, 0.09, 0.84, 0.55, -0.58, 0.21, 0.50, 0.1, -0.90\}$ and $\mathbf{x}$ of D_6 is also 2-dimensional. Figure 2.3 shows the

function surfaces when limiting the dimension to 2.

$$f_1(\mathbf{x}) = \sum_{i=1}^{7} \left[\ln^2(x_i - 2) + \ln^2(10 - x_i)\right] - \left(\prod_{i=1}^{7} x_i\right)^2$$
$$2.1 \leqslant x_i \leqslant 9.9,\ i = 1, 2, 3, \ldots, 7 \tag{2.9}$$

$$f_2(\mathbf{x}) = \sum_{j=1}^{10} \exp(x_j) \left(c_j + x_j - \ln \sum_{k=1}^{10} \exp(x_k)\right)$$
$$\mathbf{c} = [-0.6089, -17.164, -34.054, -5.914, -24.721, -14.986, -24.100,$$
$$-10.708, -26.662, -22.179]$$
$$-10 \leqslant x_i \leqslant 10 \tag{2.10}$$

$$f_3(\mathbf{x}) = x_1^2 + x_2^2 + x_1 x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 - 4(x_4 - 5)^2 + (x_5 - 3)^2$$
$$+ 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + 2(x_{10} - 7)^2 + 45$$
$$-10 \leqslant x_i \leqslant 10 \tag{2.11}$$

$$f_4(\mathbf{x}) = \sin\left(\frac{\pi x_1}{12}\right) \cos\left(\frac{\pi x_2}{16}\right)$$
$$-10 \leqslant x_1 \leqslant 10, -20 \leqslant x_2 \leqslant 20 \tag{2.12}$$

$$f_5(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{d} \alpha_i [3(1 - x_i)^2 \exp(-x_i^2 - (y_i + 1)^2) - 10(\frac{x}{5} - x_i^3 - y_i^5)$$
$$\exp(-x_i^2 - y_i^2) - \frac{1}{3}\exp(-(x_i + 1)^2 - y_i^2) + 2x_i],$$

$$\sum_{i=1}^{d} \alpha_i = 1, \text{for all } i = 1, \ldots, d,$$

$$-2 \leqslant x_i \leqslant 2, -2 \leqslant y_i \leqslant 2 \tag{2.13}$$

In the experiments, we added Gaussian noise with different levels (5%, 10%, and 20%) to the datasets to investigate and verify the robustness of the proposed methods HCM and PHCM. The signal-to-noise ratio (SNR) is defined as

$$\text{SNR} = \frac{P_{\text{signal}}}{P_{\text{noise}}} \tag{2.14}$$

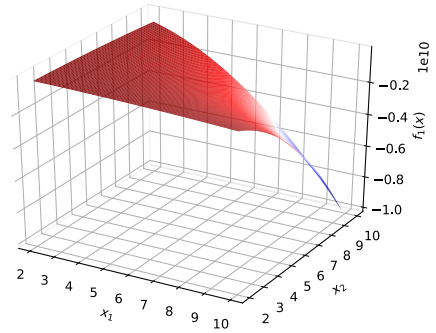
where P_{singal} is the average power of the signal and P_{noise} is the average power of the noise [16]. Figure 2.4 and Figure 2.5 show D_4 and D_6 with different noise levels.

2.5 Knot optimization for MARS using hill climbing methods

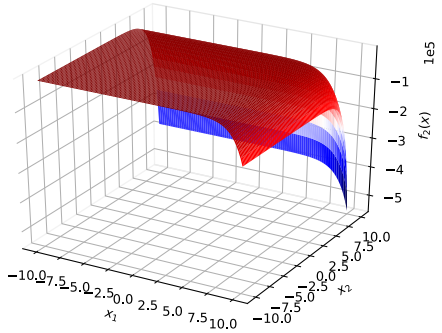
In our proposed MARS knot positioning process, we define the change in RSS as the objective function, given as

$$\Delta e = e_p - e_c,$$

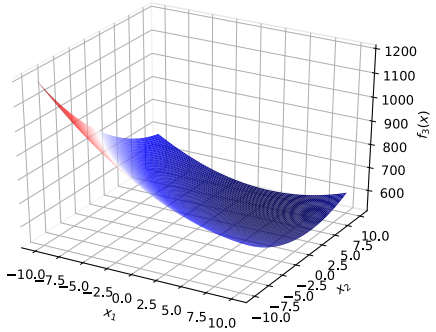
where e_p and e_c are the RSS values of the prior iteration and the current iteration, respectively. If Δe is negative, it indicates that the current MARS model is less accurate than the prior MARS model. If Δe is positive, it indi-



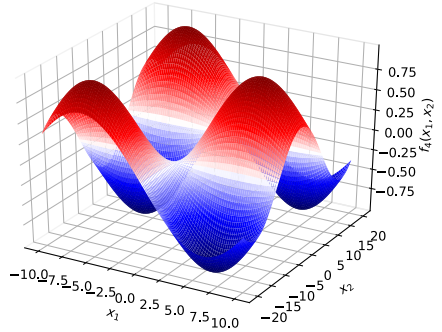
(a) $f_1(x_1, x_2)$ other $x_i = 4.0$



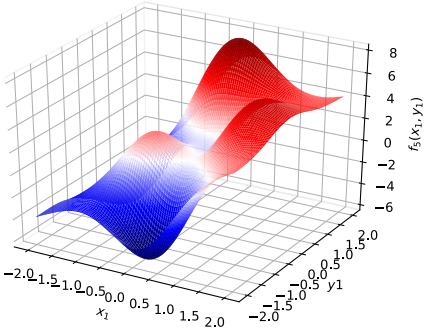
(b) $f_2(x_1, x_2)$ other $x_i = 3.0$



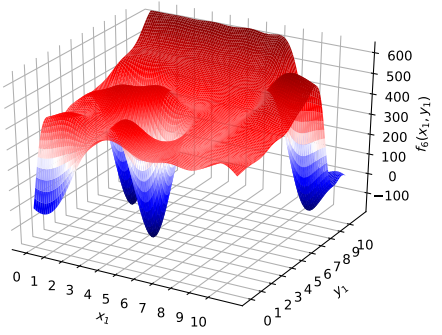
(c) $f_3(x_1, x_2)$ other $x_i = 4.0$



(d) $f_4(x_1, x_2)$

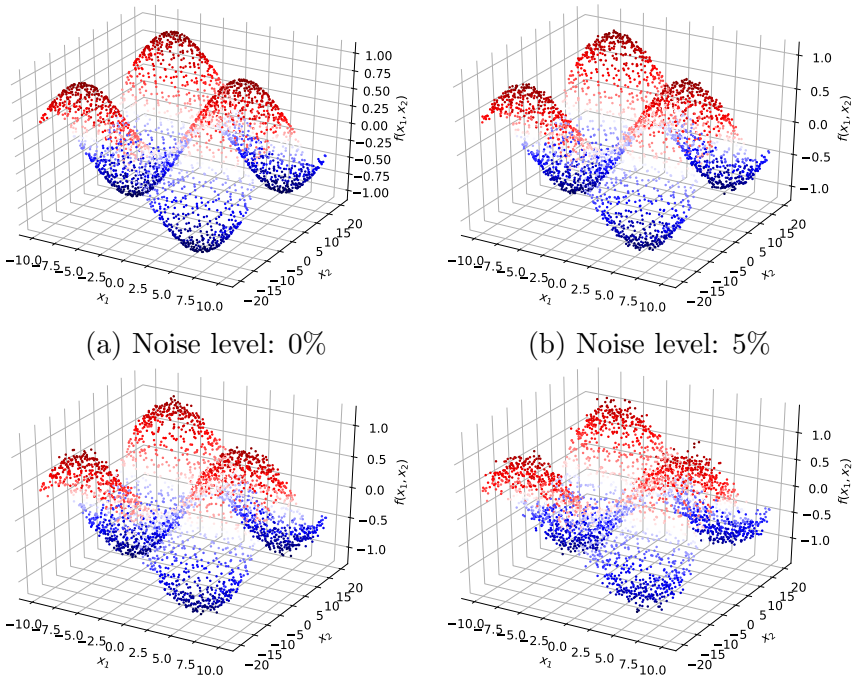


(e) $f_5(x_1, y_1)$ $d = 1$



(f) $f_6(x_1, y_1)$ $d = 1$

Figure 2.3: Surfaces of dataset functions



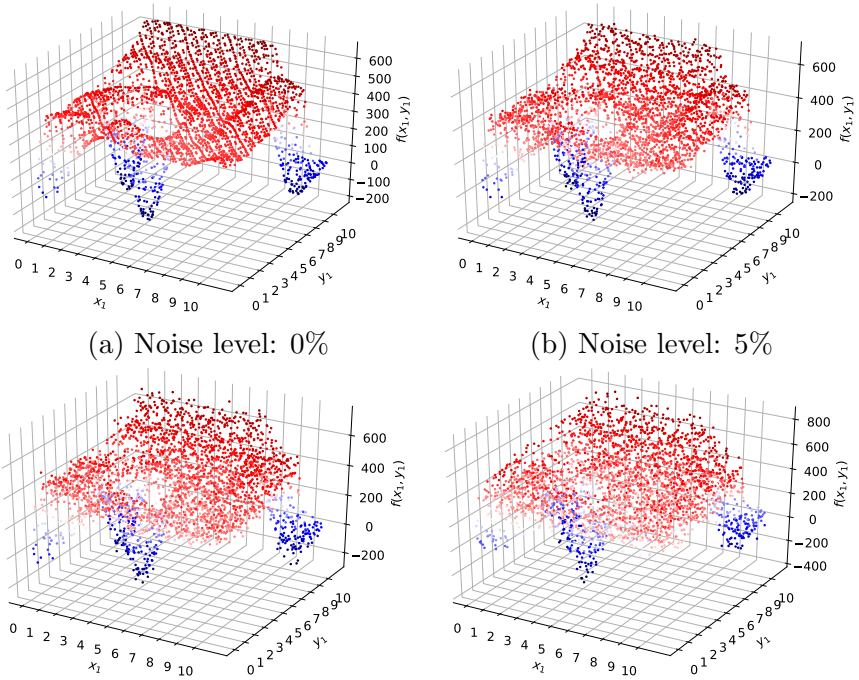
(a) Noise level: 0%

(b) Noise level: 5%

(c) Noise level: 10%

(d) Noise level: 20%

Figure 2.4: D_4 with different levels of noise



(a) Noise level: 0%

(b) Noise level: 5%

(c) Noise level: 10%

(d) Noise level: 20%

Figure 2.5: D_6 with different levels of noise

cates that the current MARS model is more accurate and is an improvement over the prior MARS model. The larger the value of Δe, the more accurate the current MARS model is. Hence, we seek to maximize Δe to find the best fitting MARS model under current settings (adding one knot to the current MARS model).

2.5.1 Exploring the change in residual sum of squares function

In the MARS knot positioning process, when we use Δe, lines 8 to 10 in in Algorithm 1 will become:

```
 8 Δe = e* − e
 9 if Δe > 0 then
10 |   e* = e, m* = m, v* = v, t* = t
11 end
```

The Δe will be calculated repeatedly for different basis functions to choose the knot with the largest Δe value. Figure 2.6 shows Δe functions of variable x_1 in MARS from subsequent iterations on a representative dataset D_L. Assume x_1 is the variable that we are considering in creating the next basis function $B_m(\mathbf{x})$. Figure 2.6a is generated when there are no basis functions in the MARS model. Figures 2.6b and 2.6e are generated when there are already two and four basis functions, respectively, in the MARS model. From Figure 2.6a to Figure 2.6e, the three local maxima move only a little, and the global maximum is almost the same position, which is around 0.25. The principle that the local maxima of Δe functions move very little from iteration to iteration also applies

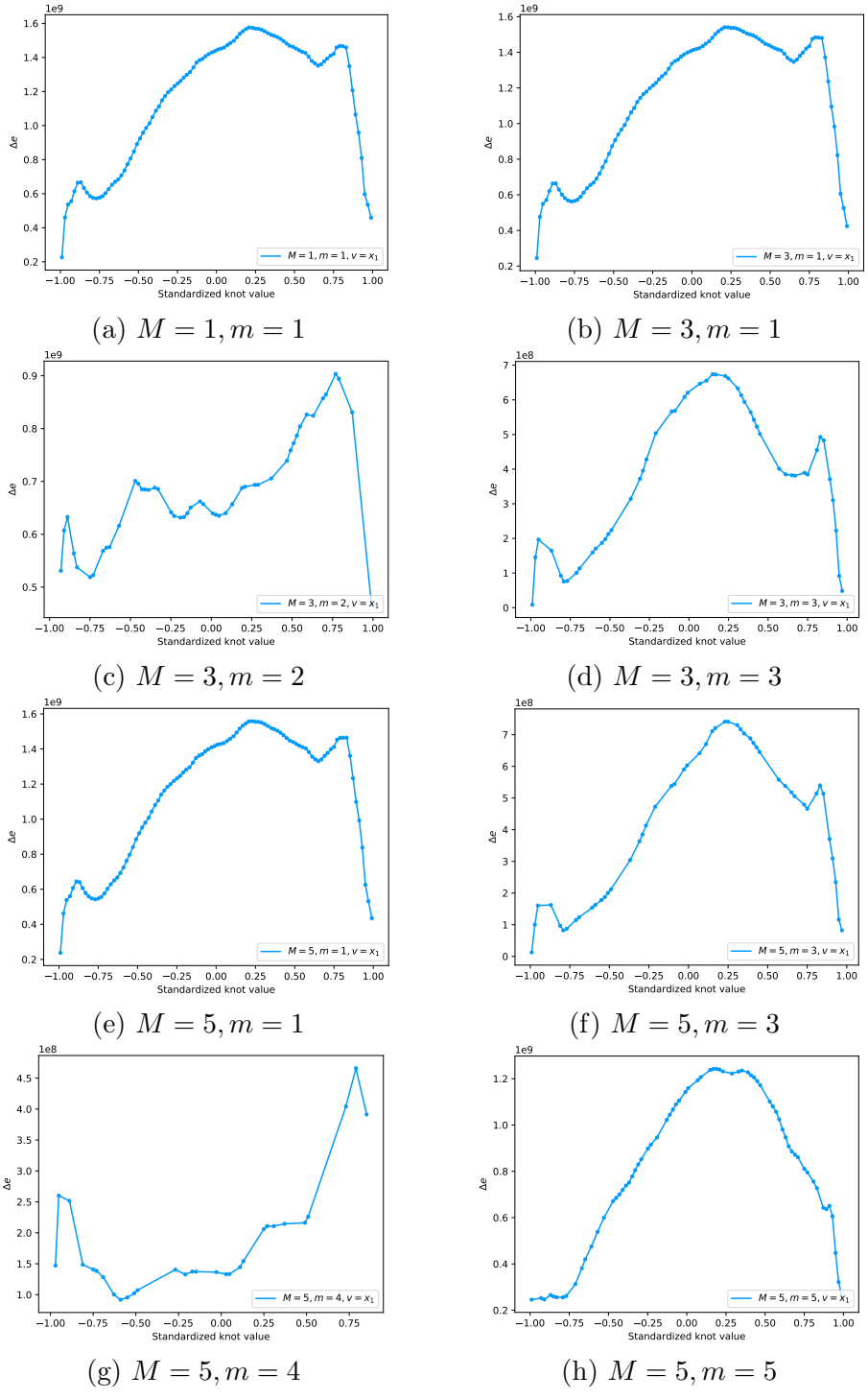
to other cases. We refer to these local maxima as *key knots*, and we will use this principle in our new knot positioning methods.

2.5.2 Hill climbing method

In this section, we introduce the hill climbing method for MARS knot positioning [17]. If a function is concave, then hill climbing will find a global maximum, if one exists. However, the Δe function may not be concave, so we require multiple starting points to get closer to the global maximum, as shown in Figure 2.7.

Figure 2.7 shows how the hill climbing method works on an example Δe function, where the vertical axis is the Δe value and the horizontal axis is the knot value. Suppose S_0, S_1, and S_2 are three candidate knot positions that are arbitrarily chosen from the candidate knot set (line 5 in Algorithm 1). If we start from S_0, S_1, and S_2 and try to maximize Δe, then we will end with knot values M_0, M_1, and M_2, respectively. Only the knot values in $[S_0, M_0]$, $[S_1, M_1]$ and $[S_2, M_2]$ will be traversed, and the other knot values in the domain of Δe will be ignored, so using hill climbing methods will speed up the knot positioning process by reducing the search process.

The starting points play an important role in the hill climbing method, which heavily affects the convergence speed and the last achieved optimum value. If the starting points are very close to a local maximum, the optimization process will end up at a local optimum, as shown in Figure 2.7 where we are trying to maximize Δe. Fortunately, by exploring Δe functions of different datasets, we find that the current key knots move a little from the prior



(a) $M = 1, m = 1$ (b) $M = 3, m = 1$

(c) $M = 3, m = 2$ (d) $M = 3, m = 3$

(e) $M = 5, m = 1$ (f) $M = 5, m = 3$

(g) $M = 5, m = 4$ (h) $M = 5, m = 5$

Figure 2.6: Explore the change in the residual sum of squares (Δe) function for $v = x_1$

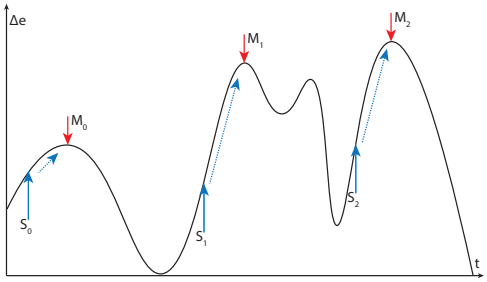


Figure 2.7: Illustration of the hill climbing method

key knots. Intuitively, we can use the key knots from a prior iteration as the starting points of the current iteration. By doing experiments on different datasets, we find that it works the same way on other datasets. This pattern of the key knots' changes can be helpful when a basis function is added using that x-variable.

2.5.3 Hill climbing method without using prior change in RSS information for MARS knot positioning

The first new method we propose for MARS knot positioning is a general hill climbing method with multiple starting points, called HCM. The HCM algorithm starts with multiple starting points and converges to the local maxima of Δe. Then the knot with the largest Δe from the local maxima is chosen as the new knot to be added to the MARS model.

Algorithm 2 shows the HCM knot positioning algorithm. The initial step in line 2 sorts the candidate knots in ascending order, and the knots are referenced by their ordered knot index. An positive integer step size r is defined to increment the knot index, which allows the algorithm to traverse the candidate

Knot index	1	2	3	4	5	6	7	8	9	10
Knot value	-0.81	-0.7	-0.6	-0.32	0.01	0.23	0.46	0.55	0.68	0.76
Δe	-81	700	1600	320	100	233	461	556	889	770

Figure 2.8: Illustration of candidate knots

knots. As recommended by Friedman [1], candidate knots are located only at data values. Figure 2.8 is an illustration of candidate knots for x_1. If the current knot index is 4 (knot value, -0.32) and the next knot index is 6 (knot value, 0.23), then $r = |6 - 4| = 2$. When r takes a large number, the knot positioning process will converge fast, but it is not stable because it may skip and miss an optimal knot. When r takes a small number, the knot selection process will converge slowly but is stable. Line 5 in Algorithm 2 follows the original MARS algorithm to define the potential candidate knot set for $B_m(\mathbf{x}_j)$ for the v-th input variable. In Algorithm 2, we refer to this set as Φ. Let $\Phi = \{-0.81, -0.7, -0.6, -0.32, 0.01, 0.23, 0.46, 0.55, 0.68, 0.76\}$ as shown in Figure 2.8. The starting knot set is $\Phi_{\mathbf{S}}$, where $\mathbf{S}$ is the knot index set of the starting knots, and we generate $\Phi_{\mathbf{S}}$ by taking equally indexed knots for a given starting knot number. As shown in Figure 2.8, if the starting knot number is 3, then $\mathbf{S}$ can be $\{1, 5, 9\}$ and $\Phi_{\mathbf{S}}$ is $\{-0.81, 0.01, 0.68\}$.

Lines 8 to 30 conduct HCM, which starts from each starting knot value in $\Phi_{\mathbf{S}}$. Line 9 obtains a starting knot value t_s, and line 10 calculates the new MARS model with two new basis functions by using the new knot value t_s. Line 11 calculates the e_s value, where e_s is the RSS value for the new MARS model by using knot value t_s. From lines 12 to 20, knots are traversed to the left of the starting knot, while from lines 21 to 29, knots are traversed to the right of the starting knot. As illustrated in Figure 2.8, if t_s is 0.01,

the knots to the left are $\{-0.7, -0.6, -0.32\}$ and the knots to the right are $\{0.23, 0.46, 0.55\}$. In line 12, when traversing knots to the left of t_s, the knot index s_- is initialized to s, and the current e for knot $\Phi[s_-]$ is e_c. In line 14, the e value e_c for the prior knot becomes the prior e value e_P for the current knot. Line 15 updates the information on the best knot. Line 16 moves the current knot index to the left by r and updates the current knot value t to $\Phi[s_-]$. Lines 17 and 18 update the MARS model with the current knot and calculate the e value e_c for the current knot. Line 19 calculates the decrease in e value Δe. If Δe is greater than a predefined small positive scalar ϵ, the current knot index will move to the left by r and repeat line 14 to 19 again. Otherwise, the algorithm will stop traversing to the left and will begin traversing to the right of the starting knot $\Phi[s]$, and in this case, the process will skip a part of knots and save time. Lines 22 to 29 traverse knots to the right of the starting knot $\Phi[s]$. The knot sets Φ_S divides the whole searching space into intervals, and if a knot t has already been traversed, the current search stops.

2.5.4 Hill climbing method using prior change in RSS information for MARS knot positioning

In HCM, all candidate knots are equally likely to be chosen for the starting point set. As was shown earlier in Figure 2.6 shows that the local optima do not move much from iteration to iteration and the local optima should be considered with much higher priority [18]. The second new method is a hill climbing method using the prior Δe information (PHCM), where the starting point set consists of the key knots from the prior iteration. By exploring the

Algorithm 2: HCM method for MARS knot positioning

Data: $\mathbf{x}, y, M_{\max}, \epsilon, r$
Result: MARS regression model $\hat{f}(\mathbf{x})$

$B_1(\mathbf{x}) = 1, M = 1, e^* = \infty$
sort $\{\mathbf{x}_{j,v}\} \rightarrow \{\mathbf{x}_{(j),v}\}$ **ascending**
while $M < M_{max}$ **do**
  for $m = 1\, to\, M$ **do**
    for $v \notin \{v(k,m)\}_{k=1}^{K_m}$ **do**
      $\Phi = \{\mathbf{x}_{j,v} | B_m(\mathbf{x}_j) > 0\}_{j=1}^{N}$
      random $\Phi_{\mathbf{S}} \subseteq \Phi$
      foreach $t_s \in \Phi_{\mathbf{S}}$ **do**
        $t = t_s$ $(t_s = \Phi[s])$
        $\hat{f} = \sum_{i=1}^{M-1} a_i B_i(\mathbf{x}) + a_{M+1} B_m(\mathbf{x})[+(x_v - t)]_+ + a_{M+2} B_m(\mathbf{x})[-(x_v - t)]_+$
        $e_s = \min_{a_1,\ldots,a_{M+2}} e(\hat{f})$
        $s_- = s, e_c = e_s$
        do
          $e_p = e_c$
          if $e_p < e^*$ **then** $e^* = e_p, m^* = m, v^* = v, t^* = t$
          $s_- = s_- - r, t = \Phi[s_-]$
          $\hat{f} = \sum_{i=1}^{M-1} a_i B_i(\mathbf{x}) + a_{M+1} B_m(\mathbf{x})[+(x_v - t)]_+ + a_{M+2} B_m(\mathbf{x})[-(x_v - t)]_+$
          $e_c = \min_{a_1,\ldots,a_{M+2}} e(\hat{f})$
          $\Delta e = e_p - e_c$
        while $\Delta e > \epsilon$
        $t = t_s, s_+ = s, e_c = e_{t_s}$
        do
          $e_p = e_c$
          if $e_p < e^*$ **then** $e^* = e_p, m^* = m, v^* = v, t^* = t$
          $s_+ = s_+ + r, t = \Phi[s_+]$
          $\hat{f} = \sum_{i=1}^{M-1} a_i B_i(\mathbf{x}) + a_{M+1} B_m(\mathbf{x})[+(x_v - t)]_+ + a_{M+2} B_m(\mathbf{x})[-(x_v - t)]_+$
          $e_c = \min_{a_1,\ldots,a_{M+2}} e(\hat{f})$
          $\Delta e = e_p - e_c$
        while $\Delta e > \epsilon$
      end
    end
  end
  $B_{M+1}(\mathbf{x}) = B_{m^*}(\mathbf{x})[+(x_{v^*} - t^*)]_+$
  $B_{M+2}(\mathbf{x}) = B_{m^*}(\mathbf{x})[-(x_{v^*} - t^*)]_+$
  $M = M + 2$
end

Δe function in Section 2.5.2, we find that local maxima from the prior iteration are usually near those of the current iteration, so we expect PHCM to converge faster than HCM.

The difference between HCM and PHCM is how to determine $\Phi_{\mathbf{S}}$. In HCM, $\Phi_{\mathbf{S}}$ is generated by taking equally indexed knots for all iterations in Algorithm 2 line 7. In PHCM, for the first iteration, all candidate knots are chosen as $\Phi_{\mathbf{S}}$, and for the other iterations, $\Phi_{\mathbf{S}}$ is the local maxima set of Δe identified in the prior iteration. For example as shown in Figure 2.8, in the first iteration for x_1,

$$\begin{aligned} &\mathbf{S} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}, \\ &\Phi_{\mathbf{S}} = \{-0.81, -0.7, -0.6, -0.32, 0.01, 0.23, 0.46, 0.55, 0.68, 0.76\}, \end{aligned} \tag{2.15}$$

and for the second iteration for x_1,

$$\begin{aligned} &\mathbf{S} = \{3, 9\}, \\ &\Phi_{\mathbf{S}} = \{-0.6, 0.68\}. \end{aligned} \tag{2.16}$$

In this way, PHCM converges faster to the local maxima, so PHCM has superiority in dealing large datasets.

2.6 Experiments and results

In this section, we test the MARS knot selection method from Chen et al. [10] and our new methods, HCM and PHCM, on different datasets with varying noise levels.

2.6.1 Exploration of candidate knot numbers

In this section, the MARS knot selection method from Chen et al. [10] (CM), HCM, and PHCM methods are applied to six datasets under different candidate knot number settings, 10, 30, 50, 100, 200, 500 and 1000. Table 2.1 summarizes the training and testing R^2 results on dataset D_1 under different candidate knot number settings, 10, 30, 50, 100, 200, 500 and 1000. Let R_P^2, R_H^2, and R_C^2 be the coefficients of determination for the PHCM method, the HCM method, and the knot positioning method from Chen et al. [10], respectively, where a higher R^2 indicated a better fit to the data. The number of candidate knot number is denoted as N_k.

From Table 2.1, we can see that as the candidate knot number increases, the R^2 value is going up. The table also shows there is no significant R^2 difference between training and testing dataset, so overfitting is not a problem. However, the computational time is also going up with the candidate knot number increasing. Under the same candidate knot number settings, the R^2 values for the CM method, the HCM method, and the PHCM method are almost the same.

Let T_C denote the computational time for CM method, T_H for HCM method and T_P for PHCM method. The computational time ratio of three methods are defined as follows:

$$\text{Computation time ratio of CM} = \frac{T_C}{T_C} = 1 \tag{2.17}$$

$$\text{Computation time ratio of HCM} = \frac{T_H}{T_C} \tag{2.18}$$

Table 2.1: R^2 comparison on dataset D_1 over different candidate knot numbers: training vs testing

Noise	N_k		10	30	50	100	200	500	1000
0%	Train	R_C^2	0.769	0.809	0.860	0.871	0.884	0.920	0.941
		R_H^2	0.781	0.799	0.846	0.860	0.882	0.912	0.938
		R_P^2	0.769	0.805	0.860	0.870	0.884	0.920	0.940
	Test	R_C^2	0.738	0.780	0.832	0.840	0.855	0.892	0.913
		R_H^2	0.752	0.769	0.817	0.829	0.852	0.883	0.910
		R_P^2	0.738	0.775	0.832	0.841	0.855	0.892	0.914
5%	Train	R_C^2	0.769	0.825	0.834	0.854	0.880	0.895	0.909
		R_H^2	0.760	0.814	0.830	0.842	0.862	0.875	0.902
		R_P^2	0.769	0.816	0.832	0.852	0.880	0.895	0.906
	Test	R_C^2	0.742	0.799	0.808	0.829	0.856	0.872	0.886
		R_F^2	0.734	0.789	0.806	0.817	0.838	0.851	0.877
		R_P^2	0.742	0.790	0.807	0.827	0.856	0.872	0.883
10%	Train	R_C^2	0.769	0.816	0.831	0.843	0.853	0.869	0.891
		R_H^2	0.766	0.813	0.825	0.836	0.847	0.861	0.875
		R_P^2	0.769	0.804	0.831	0.838	0.853	0.869	0.890
	Test	R_C^2	0.750	0.794	0.812	0.824	0.834	0.851	0.873
		R_F^2	0.746	0.790	0.805	0.816	0.827	0.842	0.856
		R_P^2	0.750	0.785	0.812	0.819	0.834	0.851	0.870
20%	Train	R_C^2	0.761	0.792	0.802	0.825	0.834	0.860	0.887
		R_F^2	0.759	0.786	0.800	0.816	0.830	0.857	0.884
		R_P^2	0.761	0.792	0.802	0.822	0.834	0.860	0.887
	Test	R_C^2	0.744	0.776	0.794	0.808	0.817	0.844	0.872
		R_F^2	0.741	0.768	0.793	0.798	0.814	0.840	0.870
		R_P^2	0.744	0.776	0.794	0.805	0.817	0.844	0.872

$$\text{Computation time ratio of PHCM} = \frac{T_P}{T_C}. \tag{2.19}$$

The computational time of CM method is the benchmark. If the computational time is less than 1, it implies that the methods uses less computational time than CM method.

Figures 2.9, 2.10 and 2.11 summarize the computational time ratios of three methods on datasets D_6, D_1 and D_5 under different candidate knot number settings. The input variable $\boldsymbol{x}$ for D_6 is 2 dimensional, $\boldsymbol{x}$ for D_1 is 7 dimensional and $\boldsymbol{x}$ for D_5 is 20 dimensional. Under most cases, the HCM and PHCM methods used less computational time than CM method. The ratio of HCM over different candidate knot numbers remained relatively stable compared to the ratio of PHCM, around 0.60 to 0.70, which indicates only 60% to 70% of the computational time of CM method was used in HCM method. The ratio of PHCM on D_1 dropped dramatically with increasing candidate knot numbers from 0.80 to 0.25, which indicates that the PHCM method used about 80% of the computational time of CM method when the candidate knot number was 10, and 25% of the CM computational time when the candidate knot number was 1000. The ratio for PHCM is going down when the candidate knot number increases, which means PHCM is more computationally efficient when dealing with a large size dataset. The figures also show that the proposed methods HCM and PHCM can perform very well with different levels of noise.

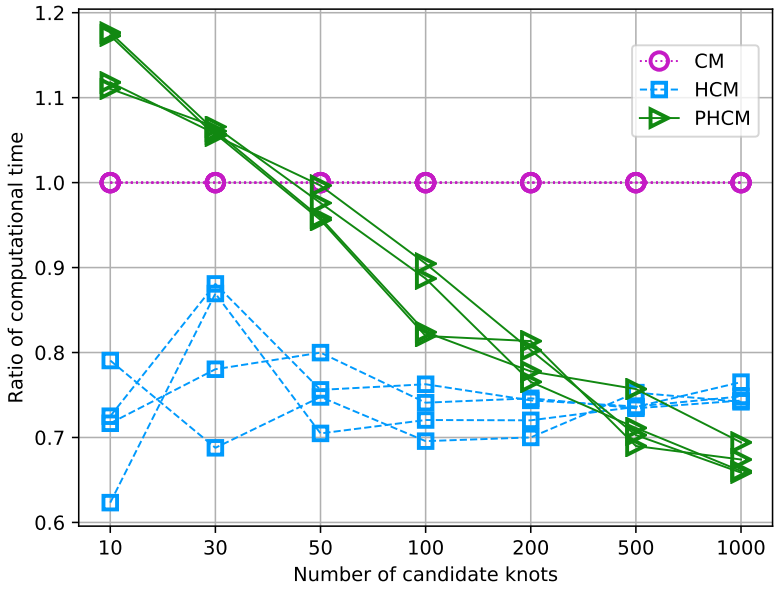


Figure 2.9: Computational time ratios of three methods on D_6 under different knot number settings with four noise levels: $\boldsymbol{x}$ is 2 dimensional

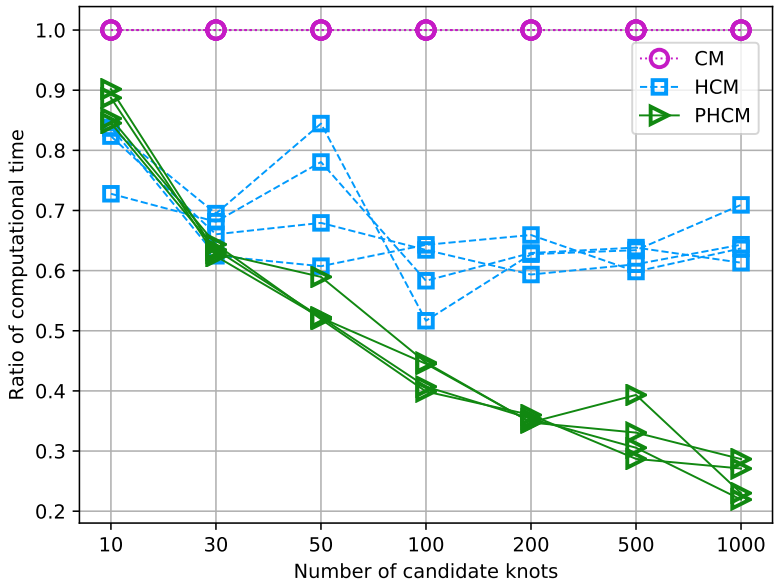


Figure 2.10: Computational time ratio of three methods on D_1 under different knot number settings with four noise levels: $\boldsymbol{x}$ is 7 dimensional

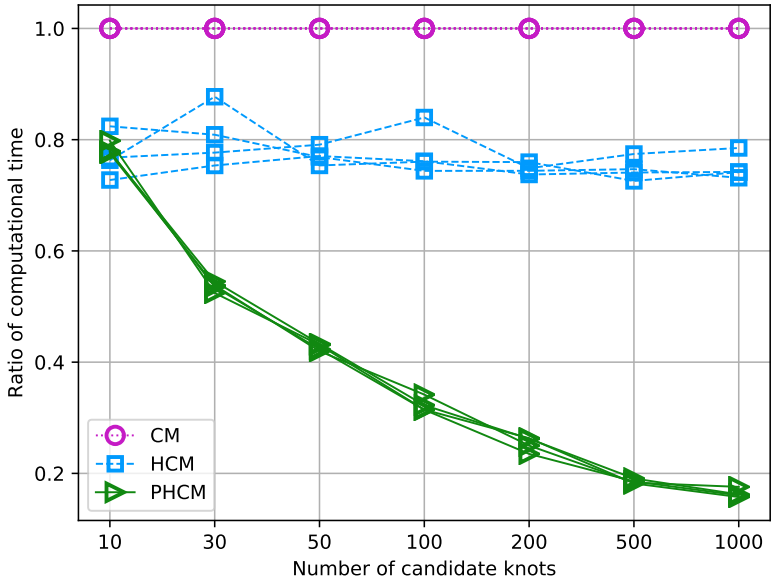


Figure 2.11: Computational time ratios of three methods on D_5 under different knot number settings with four noise levels: $\boldsymbol{x}$ is 20 dimensional

2.6.2 Exploration of different datasets

In this section, we tested our methods on six different datasets with the candidate knot number setting being 1000. The CM method, the HCM method, and the PHCM method are used on these six datasets.

Table 2.2 summarizes the R^2 values of three methods on six different datasets under four levels of noises. Under the same settings, the final achieved R^2 are almost the same. It also shows there is little difference in R^2 between the training set and the testing set, so no overfitting is again not a problem..

Figure 2.12 is the comparison of the computational time ratios of three methods on six different datasets. Comparing datasets with different dimensions, we saw that for datasets with 7, 10, 10, and 20 dimensions, the PHCM method has dramatically lower computational time ratio than HCM method.

Table 2.2: R^2 comparison on six different datasets: training vs testing

Noise	$f(\mathbf{x})$		D_1	D_2	D_3	D_4	D_5	D_6
	N_k		1000	1000	1000	1000	1000	1000
0%	Train	R^2_O	0.941	0.999	0.999	0.999	0.944	0.971
		R^2_H	0.938	0.999	0.999	0.999	0.945	0.971
		R^2_P	0.940	0.999	0.999	0.999	0.943	0.971
	Test	R^2_O	0.913	0.994	0.999	0.998	0.940	0.946
		R^2_H	0.910	0.994	0.999	0.998	0.936	0.946
		R^2_P	0.914	0.999	0.999	0.998	0.932	0.946
5%	Train	R^2_O	0.909	0.995	0.993	0.996	0.935	0.952
		R^2_H	0.902	0.995	0.993	0.996	0.935	0.953
		R^2_P	0.906	0.995	0.993	0.996	0.940	0.952
	Test	R^2_O	0.886	0.991	0.992	0.995	0.940	0.929
		R^2_H	0.877	0.991	0.999	0.995	0.940	0.927
		R^2_P	0.883	0.998	0.999	0.995	0.938	0.929
10%	Train	R^2_O	0.891	0.985	0.973	0.991	0.933	0.906
		R^2_H	0.875	0.985	0.973	0.991	0.933	0.906
		R^2_P	0.890	0.985	0.973	0.991	0.933	0.906
	Test	R^2_O	0.873	0.989	0.953	0.997	0.918	0.897
		R^2_H	0.856	0.989	0.953	0.997	0.918	0.897
		R^2_P	0.870	0.989	0.949	0.997	0.930	0.890
20%	Train	R^2_O	0.887	0.948	0.898	0.965	0.904	0.794
		R^2_H	0.884	0.948	0.898	0.965	0.904	0.794
		R^2_P	0.887	0.948	0.898	0.965	0.903	0.794
	Test	R^2_O	0.872	0.955	0.895	0.950	0.909	0.778
		R^2_H	0.870	0.955	0.896	0.950	0.908	0.778
		R^2_P	0.872	0.951	0.897	0.950	0.910	0.778

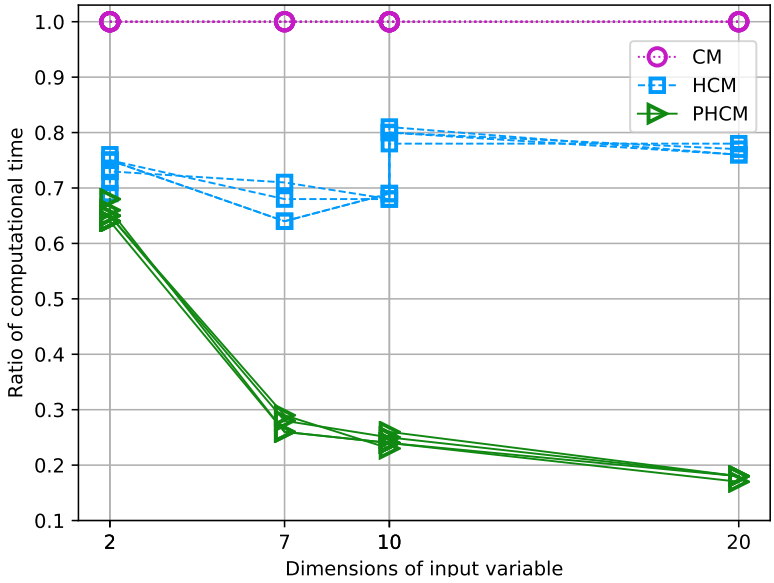


Figure 2.12: Computational time ratios of three methods on six datasets with different dimensions under four noise levels

For datasets with 2 dimensions, the differences in the ratio are not as dramatic as those for high dimensional datasets. The phenomenon indicates that PHCM should be the preferred method for high-dimensional data.

2.7 Conclusion

In this paper, we proposed two new methods for MARS knot positioning, the hill climbing method (HCM) and the hill climbing method using key knots. The HCM and PHCM achieved a reduction in computational time compared to CM, while maintaining similar quality of fit. The PHCM achieved the most significant savings with over 80% reduction in computational time for the higher-dimensional data sets. By using different datasets with different noise levels, we show that PHCM and HCM are robust dealing with noisy

datasets.

Bibliography

[1] Jerome H Friedman. Multivariate adaptive regression splines. *The annals of statistics*, pages 1–67, 1991.

[2] JR Leathwick, D Rowe, J Richardson, Jane Elith, and T Hastie. Using multivariate adaptive regression splines to predict the distributions of new zealand's freshwater diadromous fish. *Freshwater Biology*, 50(12):2034–2052, 2005.

[3] JR Leathwick, J Elith, and T Hastie. Comparative performance of generalized additive models and multivariate adaptive regression splines for statistical modelling of species distributions. *Ecological modelling*, 199(2): 188–196, 2006.

[4] Tian-Shyug Lee, Chih-Chou Chiu, Yu-Chao Chou, and Chi-Jie Lu. Mining the customer credit using classification and regression tree and multivariate adaptive regression splines. *Computational Statistics & Data Analysis*, 50(4):1113–1130, 2006.

[5] Joel Deichmann, Abdolreza Eshghi, Dominique Haughton, Selin Sayek, and Nicholas Teebagy. Application of multiple adaptive regression splines (mars) in direct response modeling. *Journal of Interactive Marketing*, 16 (4):15–27, 2002.

[6] Zehua Yang, Victoria CP Chen, Michael E Chang, Melanie L Sattler, and Aihong Wen. A decision-making framework for ozone pollution control. *Operations Research*, 57(2):484–498, 2009.

[7] Christian Conoscenti, Valerio Agnesi, Mariaelena Cama, Nathalie Alamaru Caraballo-Arias, and Edoardo Rotigliano. Assessment of gully erosion susceptibility using multivariate adaptive regression splines and accounting for terrain connectivity. *Land degradation & development*, 29(3): 724–736, 2018.

[8] Sanjiban Sekhar Roy, Reetika Roy, and Valentina E Balas. Estimating heating load in buildings using multivariate adaptive regression splines, extreme learning machine, a hybrid model of mars and elm. *Renewable and Sustainable Energy Reviews*, 82:4256–4268, 2018.

[9] Salim Heddam and Ozgur Kisi. Modelling daily dissolved oxygen concentration using least square support vector machine, multivariate adaptive regression splines and m5 model tree. *Journal of hydrology*, 559:499–509, 2018.

[10] Victoria CP Chen, David Ruppert, and Christine A Shoemaker. Applying experimental design and regression splines to high-dimensional continuous-state stochastic dynamic programming. *Operations Research*, 47(1):38–53, 1999.

[11] Elcin Kartal Koc and Cem Iyigun. Restructuring forward step of mars

algorithm using a new knot selection procedure based on a mapping approach. *Journal of Global Optimization*, 60(1):79–102, 2014.

[12] Satoshi Miyata and Xiaotong Shen. Free-knot splines and adaptive knot selection. *Journal of the Japan Statistical Society*, 35(2):303–324, 2005.

[13] P Bratley and BL Fox. Implementing sobols quasirandom sequence generator (algorithm 659). *ACM Transactions on Mathematical Software*, 29 (1):49–57, 2003.

[14] Feng Liu and Zhifang Wang. A novel adaptive genetic algorithm for wine farm layout optimization. In *Power Symposium (NAPS), 2017 North American*, pages 1–6. IEEE, 2017.

[15] Manuel Laguna and Rafael Martí. Experimental testing of advanced scatter search designs for global optimization of multimodal functions. *Journal of Global Optimization*, 33(2):235–255, 2005.

[16] Don H Johnson. Signal-to-noise ratio. *Scholarpedia*, 1(12):2088, 2006.

[17] Kiarash Mahdavi, Mark Harman, and Robert M Hierons. A multiple hill climbing approach to software module clustering. In *Software Maintenance, 2003. ICSM 2003. Proceedings. International Conference on*, pages 315–324. IEEE, 2003.

[18] Xinglong Ju, Youxiang Duan, Cheng Ma, and Haiyan Ju. Predicting service execution time towards a runtime monitoring approach. In *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2013 International Conference on*, pages 221–224. IEEE, 2013.

Appendix

2.A Other results tables and charts

Table 2.A.1 summarizes the training results on dataset D_1 under different candidate knot number settings, 10, 30, 50, 100, 200, 500 and 1000. Define N_k to be the number of candidate knots. Let N_C be the total number of knots in which Δe was calculated using the CM, N_H be that using the HCM, and N_P be that using the PHCM, where fewer calculated Δe values usually result in a lower computational time. Let R_P be the ratio of N_P to N_C, and R_H be the ratio of N_H to N_C, where a lower ratio indicates lower computational effort. Let R_P^2, R_H^2, and R_C^2 be the coefficients of determination for the PHCM, the HCM, and the CM, respectively, where a higher R^2 indicated a better fit to the data. Let T be the computational time in seconds of the MARS algorithm with T_P for the PHCM, T_H for the HCM, and T_C for the CM.

We also tested our methods on six different datasets with the candidate knot number setting being 1000. The CM method, the HCM method, and the PHCM method are used on these six datasets.

Table 2.A.2 summarizes the training results on six different datasets with

Table 2.A.1: Comparison of three methods on dataset D_1 over different candidate knot numbers: training results

Noise	N_k	10	30	50	100	200	500	1000
0%	N_C	17,197	53,675	93,610	181,336	370,894	1,086,174	2,090,062
	N_H	12,521	33,630	58,586	119,062	242,323	656,259	1,339,374
	N_P	13,032	31,134	46,381	69,180	130,477	320,969	590,287
	R_H	0.73	0.63	0.63	0.66	0.65	0.60	0.64
	R_P	0.76	0.58	0.50	0.38	0.35	0.30	0.28
	R^2_C	0.769	0.809	0.860	0.871	0.884	0.920	0.941
	R^2_H	0.781	0.799	0.846	0.860	0.882	0.912	0.938
	R^2_P	0.769	0.805	0.860	0.870	0.884	0.920	0.940
	R^2_{aC}	0.744	0.790	0.845	0.853	0.873	0.913	0.936
	R^2_{aH}	0.757	0.778	0.830	0.843	0.872	0.905	0.932
	R^2_{aP}	0.744	0.785	0.845	0.842	0.873	0.913	0.929
	T_C	3.40	7.94	13.27	24.36	46.60	146.38	259.29
	T_H	2.85	4.96	8.06	15.66	30.72	87.61	165.16
	T_P	2.90	5.11	6.89	9.72	16.77	42.01	70.25
5%	N_C	17,282	56,015	86,755	192,700	348,157	869,325	1,988,368
	N_H	12,610	35,746	56,893	125,606	218,216	535,707	1,266,933
	N_P	13,110	31,609	42,310	82,831	121,907	257,309	518,964
	R_H	0.73	0.64	0.66	0.65	0.63	0.62	0.64
	R_P	0.76	0.56	0.49	0.43	0.35	0.30	0.26
	R^2_C	0.769	0.825	0.834	0.854	0.880	0.895	0.909
	R^2_H	0.760	0.814	0.830	0.842	0.862	0.875	0.902
	R^2_P	0.769	0.816	0.832	0.852	0.880	0.895	0.906
	R^2_{aC}	0.743	0.807	0.818	0.835	0.868	0.875	0.900
	R^2_{aH}	0.740	0.794	0.811	0.831	0.846	0.865	0.881
	R^2_{aP}	0.743	0.801	0.815	0.833	0.865	0.874	0.897
	T_C	3.46	8.51	11.54	26.94	45.66	102.34	259.35
	T_H	2.85	5.62	7.84	17.09	27.10	62.49	166.78
	T_P	3.07	5.31	6.02	11.99	15.88	40.25	59.66
10%	N_C	17,222	52,221	85,520	188,521	381,650	879,351	1,336,330
	N_H	13,084	33,790	53,963	113,931	253,330	585,591	909,954
	N_P	12,965	29,753	42,885	78,657	136,553	275,360	341,559
	R_H	0.76	0.65	0.63	0.60	0.66	0.67	0.68
	R_P	0.75	0.57	0.50	0.42	0.36	0.31	0.26
	R^2_C	0.769	0.816	0.831	0.843	0.853	0.869	0.891
	R^2_H	0.766	0.813	0.825	0.836	0.847	0.861	0.875
	R^2_P	0.769	0.804	0.831	0.838	0.853	0.869	0.890
	R^2_{aC}	0.744	0.796	0.814	0.826	0.838	0.857	0.880
	R^2_{aH}	0.741	0.793	0.811	0.821	0.833	0.852	0.861
	R^2_{aP}	0.744	0.784	0.814	0.823	0.838	0.857	0.878
	T_C	3.49	7.72	11.85	26.70	51.18	110.18	253.85
	T_H	2.54	5.26	9.25	15.57	32.21	70.33	155.65
	T_P	2.95	4.90	6.19	10.87	18.10	33.66	55.67
20%	N_C	17,762	50,280	84,420	172,248	354,203	799,428	1,755,070
	N_H	12,856	34,745	61,390	94,171	237,130	497,342	1,246,389
	N_P	13,976	29,537	43,982	74,961	128,663	254,856	514,569
	R_H	0.72	0.69	0.73	0.55	0.67	0.62	0.71
	R_P	0.79	0.59	0.26	0.44	0.36	0.32	0.29
	R^2_C	0.761	0.792	0.802	0.825	0.834	0.860	0.887
	R^2_H	0.759	0.786	0.800	0.816	0.830	0.857	0.884
	R^2_P	0.761	0.792	0.802	0.822	0.834	0.860	0.887
	R^2_{aC}	0.735	0.771	0.790	0.805	0.818	0.848	0.877
	R^2_{aH}	0.733	0.770	0.789	0.792	0.820	0.844	0.874
	R^2_{aP}	0.735	0.771	0.787	0.801	0.821	0.851	0.877
	T_C	3.56	7.70	10.16	23.13	47.44	83.5	199.02
	T_H	2.98	5.35	8.58	11.95	29.76	52.94	141.16
	T_P	3.21	4.85	5.99	10.34	16.48	27.61	57.05

the candidate knot number setting being 1000.

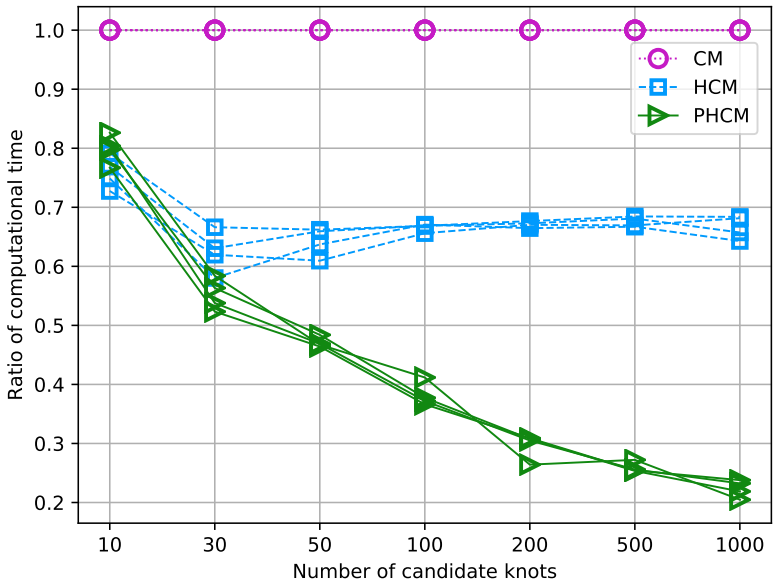


Figure 2.A.1: Computational time ratios of three methods on D_2 under different knot number settings with four noise levels: $\boldsymbol{x}$ is 10 dimensional

Table 2.A.2: Result comparison of three methods on six different datasets: training results

Noise		D_1	D_2	D_3	D_4	D_5	D_6
	N_k	1000	1000	1000	1000	1000	1000
	N_C	2,090,062	11,051,248	10,871,840	376,966	12,382,743	353,038
	N_H	1,339,374	7,581,651	8,527,864	275,526	9,603,147	265,990
	N_P	590,287	2,806,414	2,773,136	240,948	2,290,161	231,107
	R_H	0.64	0.69	0.78	0.73	0.78	0.75
	R_P	0.28	0.25	0.26	0.64	0.18	0.65
0%	R^2_C	0.941	0.999	0.999	0.999	0.944	0.971
	R^2_H	0.938	0.999	0.999	0.999	0.945	0.971
	R^2_P	0.940	0.999	0.999	0.999	0.943	0.971
	R^2_{aC}	0.936	0.999	0.999	0.999	0.938	0.970
	R^2_{aH}	0.932	0.999	0.999	0.999	0.940	0.970
	R^2_{aP}	0.929	0.999	0.999	0.999	0.938	0.970
	T_C	259.29	1149.26	960.37	30.36	1340.91	28.20
	T_H	165.16	755.43	758.20	22.48	1052.71	21.10
	T_P	70.25	273.41	224.10	19.48	235.69	19.00
	N_C	1,988,368	10,710,274	7,174,956	319,140	12,515,344	361,014
	N_H	1,266,933	7,404,639	5,800,284	220,277	9,540,538	269,215
	N_P	518,964	2,621,238	1,712,336	203,052	2,208,984	236,997
	R_H	0.64	0.69	0.81	0.69	0.76	0.75
	R_P	0.26	0.24	0.24	0.64	0.18	0.66
5%	R^2_C	0.909	0.995	0.993	0.996	0.935	0.952
	R^2_H	0.902	0.995	0.993	0.996	0.935	0.953
	R^2_P	0.906	0.995	0.993	0.996	0.940	0.952
	R^2_{aC}	0.900	0.995	0.992	0.996	0.929	0.949
	R^2_{aH}	0.881	0.995	0.992	0.996	0.929	0.950
	R^2_{aP}	0.897	0.995	0.992	0.996	0.933	0.949
	T_C	259.35	1096.25	648.71	25.68	1406.60	29.33
	T_H	166.78	749.59	574.37	17.52	1028.94	21.80
	T_P	59.66	254.78	146.67	16.47	225.57	20.36
	N_C	1,336,330	9,364,412	5,505,934	377,963	12,363,800	325,122
	N_H	909,954	6,365,090	4,421,255	268,752	9,462,023	242,482
	N_P	341,559	2,264,578	1,360,088	240,343	2,202,486	208,351
	R_H	0.68	0.68	0.80	0.71	0.77	0.75
	R_P	0.26	0.24	0.25	0.64	0.18	0.64
10%	R^2_C	0.891	0.985	0.973	0.991	0.933	0.906
	R^2_H	0.875	0.985	0.973	0.991	0.933	0.906
	R^2_P	0.890	0.985	0.973	0.991	0.933	0.906
	R^2_{aC}	0.880	0.984	0.971	0.990	0.927	0.900
	R^2_{aH}	0.861	0.984	0.971	0.990	0.927	0.900
	R^2_{aP}	0.878	0.984	0.970	0.990	0.926	0.900
	T_C	253.85	1051.80	525.21	31.62	1368.79	25.20
	T_H	155.65	676.18	436.86	22.11	1015.08	19.28
	T_P	55.67	230.08	120.25	19.78	222.06	16.64
	N_C	1,755,070	6,510,958	4,491,985	411,861	11,454,536	331,104
	N_H	1,246,389	4,458,339	3,600,931	313,356	8,692,736	242,237
	N_P	514,569	1,477,733	1,061,862	280,717	1,987,142	213,803
	R_H	0.71	0.68	0.80	0.76	0.76	0.73
	R_P	0.29	0.23	0.24	0.68	0.17	0.65
20%	R^2_C	0.887	0.948	0.898	0.965	0.904	0.794
	R^2_H	0.884	0.948	0.898	0.965	0.904	0.794
	R^2_P	0.887	0.948	0.898	0.965	0.903	0.794
	R^2_{aC}	0.877	0.942	0.887	0.963	0.894	0.781
	R^2_{aH}	0.874	0.942	0.887	0.963	0.894	0.781
	R^2_{aP}	0.877	0.942	0.887	0.963	0.893	0.781
	T_C	199.02	772.69	418.99	33.97	1243.55	26.18
	T_H	141.16	526.57	340.56	25.89	922.75	19.42
	T_P	57.05	158.24	92.90	23.50	195.84	17.24

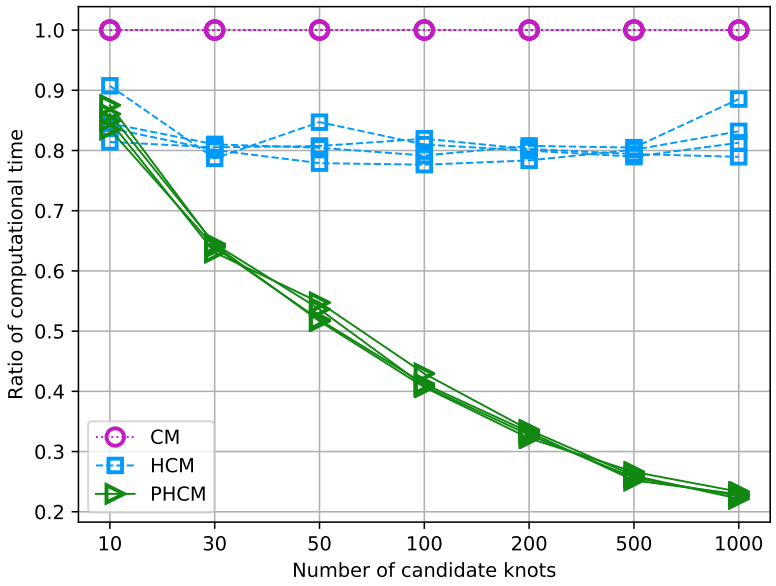


Figure 2.A.2: Computational time ratios of three methods on D_3 under different knot number settings with four noise levels: $\boldsymbol{x}$ is 10 dimensional

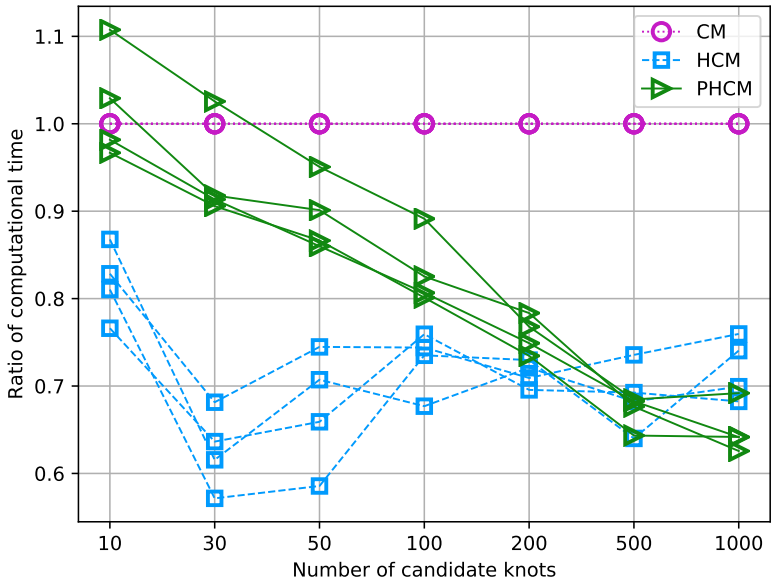


Figure 2.A.3: Computational time ratios of three methods on D_3 under different knot number settings with four noise levels: $\boldsymbol{x}$ is 2 dimensional

CHAPTER 3

GLOBAL OPTIMIZATION USING MIXED INTEGER QUADRATIC PROGRAMMING ON NON-CONVEX TWO-WAY INTERACTION TRUNCATED LINEAR MULTIVARIATE ADAPTIVE REGRESSION SPLINES

Xinglong Ju, Victoria C. P. Chen, Jay M. Rosenberger, Feng Liu

3.1 Abstract

Multivariate adaptive regression splines (MARS) is a flexible statistical modeling method that has been popular for data mining applications. MARS has also been employed to approximate unknown relationships in optimization for complex systems, including surrogate optimization, dynamic programming, and two-stage stochastic programming. Given the increasing desire to optimize real world systems, this paper presents an approach to globally optimize a MARS model that allows up to two-way interaction terms that are products of truncated linear univariate functions (TITL-MARS). Specifically, such a MARS model consists of linear and quadratic structure. This structure is exploited to formulate a mixed integer quadratic programming problem (TITL-MARS-OPT).

To appreciate the contribution of TITL-MARS-OPT, one must recognize that popular heuristic optimization approaches, such as evolutionary algorithms, do not guarantee global optimality and can be computationally slow. The use of MARS maintains the flexibility of modeling within TITL-MARS-OPT while also taking advantage of the linear modeling structure of MARS to enable global optimality. Computational results compare TITL-MARS-OPT with a genetic algorithm for two types of cases. First, a wind farm power distribution case study is described and then other TITL-MARS forms are tested. The results show the superiority of TITL-MARS-OPT over the genetic algorithm in both accuracy and computational time.

3.2 Introduction

Optimization for complex systems often involves fitting a system prediction model to estimate how a system performs and then optimizing the decisions based on the system prediction model as shown in Figure 3.1. Two major tasks in optimization of complex systems include training or meta-modeling a statistical or system model and optimizing input or decisions based on statistical model.

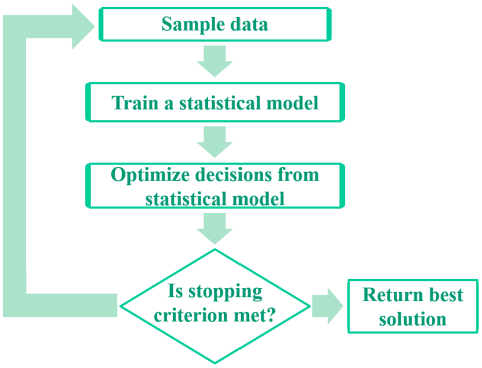


Figure 3.1: Optimization of complex systems

In real world complex systems, underlying relationships are commonly unknown and are approximated from data using empirical models. Wu et al. [1] applied support vector regression in travel time prediction and proved support vector regression was applicable in traffic data analysis. Lv et al. [2] applied a deep learning approach with autoencoders in traffic flow prediction. Ohlmacher and Davis [3] used logistic regression to generate a landslide-hazard map to predict landslide hazards. Leathwick et al. [4] used multivariate adaptive regression splines to predict the distributions of freshwater diadromous

fish.

If one seeks to optimize a complex system, the optimization method would need to be able to handle the data-driven approximation models. Given the wide range of possible approximation models, such as machine learning algorithms, the most commonly employed optimization approach in these situations is a heuristic approach, such as an evolutionary algorithm, that cannot guarantee global optimality. Rather than having the approximation model dictate the need for a heuristic optimization method, the research in this paper seeks a balance that utilizes a flexible approximation model with structure that can be exploited to enable true global optimization. In other words, the "best of both worlds" is sought, by achieving global optimality while still maintaining a flexible approximation model. The approximation model of choice in this paper is multivariate adaptive regression splines (MARS), introduced by machine learning pioneer Jerome Friedman in 1991 [5]. The structure of MARS is based on a linear statistical spline model and provides a flexible fit to data while also achieving a parsimonious model.

The desire to conduct global optimization is seen in many applications, and there are a number of approaches classified as global optimization methods [6]. The primary challenge in achieving global optimality is that many real world applications involve multiple local optima. Finding a global optimum requires sifting through the local optima and recognizing when one is suboptimal. The vast majority of applications employ heuristic search algorithms seek to overcome the challenge of local optima, but do not guarantee global optimality. Examples include heuristics based on evolutionary algorithms [7, 8, 9],

particle swarm optimization [10], the grasshopper optimization method [11], and the weighted superposition attraction method [12]. In order to guarantee global optimality, the approach in this paper takes advantage of well-known properties of mixed integer and quadratic programming (MIQP) [13].

Some recent applications in which MARS has been employed for empirical modeling include a water pollution prediction problem [14], the head load in a building [15], the estimation of landfill leachate [16], and the damage identification for web core composite bridges [17], In optimization problems, MARS has been employed as the empirical model to approximate unknown relationships in a variety of applications. For stochastic dynamic programming, the use of MARS to approximate the value function was introduced by Chen et al. in 1999 [18]. Since then, the MARS value function approximation approach has been used to numerically solve a 30-dimensional water reservoir management problem [19], a 20-dimensional wastewater treatment system [20, 21, 22], and a 524-dimensional nonstationary ground-level ozone pollution control problem [23]. In revenue management, MARS was employed to estimate upper and lower bounds for the value function of a Markov decision problem [24, 25], and MARS was used to represent the revenue function in airline overbooking optimization [26]. In two-stage stochastic programming, MARS was used to efficiently represent the expected profit function for an airline fleet assignment problem [27]. This fleet assignment research was extended to utilize a cutting plane method with MARS to conduct the optimization [28].

The contribution of this current work extends the approach of Martinez et al. [29], who developed a piece-wise linear MARS structure and formulated

a mixed integer and linear programming problem to globally optimize vehicle design parameters to improve performance in crash simulations. The piece-wise linear MARS function may be nonconvex, and the approach of Martinez et al. will yield a global optimum. However, restricting to piece-wise linear forms limits the flexibility of the empirical model. Hence, in the current work, the MARS form employed is based on the original MARS model. The primary challenge for an optimization method is handling the nonconvex MARS interaction terms, which are products of univariate terms. By restricting to two-way interactions, we can utilize quadratic programming methods. In real world applications, two-way interactions are commonly sufficient for empirical modeling [30]

In summary, the contribution of the presented approach is a MIQP global optimization method for a MARS model that allows up to two-way interaction terms that are products of truncated linear univariate functions (TITL-MARS). This approach is referred to as TITL-MARS-OPT and is compared against a genetic algorithm for two types of cases. First, a wind farm power distribution case study is described, and then other TITL-MARS forms are tested.

The rest of this paper is organized as follows. Section 3.3 describes background on TITL-MARS. Section 3.4 presents the MIQP formulation for TITL-MARS-OPT. The computational study is given in Section 3.5, and Section 3.6 concludes the paper.

3.3 Background of two-way interaction truncated linear multivariate adaptive regression splines

This section introduces the two-way interaction truncated linear MARS (TITL-MARS) model. The two-way interaction truncated linear MARS regression model with the response variable $f(\mathbf{x}_i)$ is to be built on the independent variable $\mathbf{x}_i$ and can be written in the form of the linear combination of the basis functions as [5]

$$\hat{f}(\mathbf{x}) = a_0 + \sum_{m=1}^{M} \{a_m \cdot B_m(\mathbf{x})\}. \tag{3.1}$$

The MARS model is denoted as $\hat{f}(\mathbf{x})$, and a_0 is the constant term of the model. The basis function is denoted as $B_m(\mathbf{x})$, and a_m is the coefficient of $B_m(\mathbf{x})$. The index of the basis function is denoted as m, and M is the total number of basis functions. The basis function $B_m(\mathbf{x})$ using the truncated linear term has the following form

$$B_m(\mathbf{x}) = \prod_{k=1}^{Km} [s_{k,m} \cdot (x_{v(k,m)} - t_{v(k,m)})]_+. \tag{3.2}$$

The truncated linear term is denoted as $[s_{k,m} \cdot (x_{v(k,m)} - t_{v(k,m)})]_+$, and the basis function $B_m(\mathbf{x})$ is the product of truncated linear terms. The index of the truncated linear term in $B_m(\mathbf{x})$ is denoted as k, and K_m is the total number of truncated linear terms in $B_m(\mathbf{x})$. The sign of the truncated linear term is $s_{k,m}$, which can be $+1$ or -1. The v-th component of $\mathbf{x}$ is denoted as $x_{v(k,m)}$,

and $t_{v(k,m)}$ is the corresponding knot value. *TITL-MARS is the special case of MARS in which* $K_m \leqslant 2$.

3.4 Formulation of two-way interaction truncated linear MARS using mixed integer quadratic programming

The general mixed integer quadratic programming problem [13] is given as

$$\begin{aligned} \min \quad & \tfrac{1}{2}\mathbf{z}^T\mathbf{Q}\mathbf{z} + \mathbf{c}^T\mathbf{z} \\ s.t. \quad & \mathbf{A}\mathbf{z} = \mathbf{b} \\ & \mathbf{l} \leqslant \mathbf{z} \leqslant \mathbf{u} \\ & \mathbf{z} \in \mathbb{R}^P \times \mathbb{Z}^{D-P}, \end{aligned} \tag{3.3}$$

while the two-way interaction truncated linear MARS is given in Section 3.3. In (3.3), the decision variable is $\mathbf{z}$, and the quadratic coefficients matrix is $\mathbf{Q}$. The coefficients of the linear terms in the objective function are in vector $\mathbf{c}$. The linear constraints are denoted as $\mathbf{A}\mathbf{z} = \mathbf{b}$. The lower bound and upper bound of $\mathbf{z}$ are $\mathbf{l}$ and $\mathbf{u}$, respectively. The dimension of $\mathbf{z}$ is D. There are P dimensions of real values, and $D - P$ dimensions of integers. Problem in the form 3.3 can be solved using the CPLEX solver.

The TITL-MARS optimization problem is given as follows.

$$\min \qquad \hat{f}(\mathbf{x}) = a_0 + \textstyle\sum_{m=1}^{M} \{a_m \cdot B_m(\mathbf{x})\} \tag{3.4}$$

$$\text{s.t.} \qquad \mathbf{l} \leqslant \mathbf{x} \leqslant \mathbf{u} \tag{3.5}$$

$$\mathbf{x} \in \mathbb{R}^P \times \mathbb{Z}^{D-P} \tag{3.6}$$

$$B_m(\mathbf{x}) = \prod_{k=1}^{Km} [s_{k,m} \cdot (x_{v(k,m)} - t_{v(k,m)})]_+ \tag{3.7}$$

$$[s_{k,m} \cdot (x_{v(k,m)} - t_{v(k,m)})]_+ = \max\{s_{k,m} \cdot (x_{v(k,m)} - t_{v(k,m)}), 0\} \tag{3.8}$$

The objective function (3.4) is the TITL-MARS model. The constraint set (3.5) is the boundary of $\mathbf{x}$. The constraint set (3.6) specifies the data types. Constraints (3.7) and (3.8) specify the basis functions and the truncated linear terms.

Let $\mathcal{M}$ denote an upper bound of $|x_{v(k,m)} - t_{v(k,m)}|$ and $|t_{v(k,m)} - x_{v(k,m)}|$. Let $y_{k,m}$ be an indicator variable for the nonnegativity of $s_{k,m} \cdot (x_{v(k,m)} - t_{v(k,m)})$, an let $\eta_{k,m}$ denote the univariate truncated linear function, given as

$$\eta_{k,m} = [s_{k,m} \cdot (x_{v(k,m)} - t_{v(k,m)})]_+ = \max\{s_{k,m} \cdot (x_{v(k,m)} - t_{v(k,m)}), 0\}. \tag{3.9}$$

Specifically, when $s_{k,m} \cdot (x_{v(k,m)} - t_{v(k,m)}) \geqslant 0$, $y_{k,m} = 1$ and $\eta_{k,m} = s_{k,m} \cdot (x_{v(k,m)} - t_{v(k,m)})$, otherwise $y_{k,m} = 0$ and $\eta_{k,m} = 0$.

The TITL-MARS optimization problem can be formulated into a general mixed integer quadratic programming problem as follows.

$$\min \qquad a_0 + \sum_{m=1}^{M} \left\{ a_m \cdot \prod_{k=1}^{Km} \eta_{k,m} \right\} \tag{3.10}$$

$$\text{s.t.} \quad s_{k,m} \cdot (x_{v(k,m)} - t_{v(k,m)}) \leqslant \eta_{k,m} \leqslant s_{k,m} \cdot (x_{v(k,m)} - t_{v(k,m)}) + \mathcal{M} \cdot (1 - y_{k,m}),$$

$$\forall k = 1, \ldots, K_m, \forall m = 1, \ldots, M \tag{3.11}$$

$$0 \leqslant \eta_{k,m} \leqslant \mathcal{M} \cdot y_{k,m}, \forall k = 1, \ldots, K_m, \forall m = 1, \ldots, M \tag{3.12}$$

$$\mathbf{l} \leqslant \mathbf{x} \leqslant \mathbf{u} \tag{3.13}$$

$$\mathbf{x} \in \mathbb{R}^P \times \mathbb{Z}^{D-P} \tag{3.14}$$

$$\eta_{k,m} \in \mathbb{R}, \forall k = 1, \ldots, K_m, \forall m = 1, \ldots, M \tag{3.15}$$

$$y_{k,m} \in \mathbb{B}, \forall k = 1, \ldots, K_m, \forall m = 1, \ldots, M. \tag{3.16}$$

The objective (3.10) is the TITL-MARS model. Equations (3.11) - (3.16) formulate the basis functions into linear constraints and specifying the boundaries and data types.

TITL-MARS-OPT is an optimization process as shown in Figure 3.1. The process has two steps. The first step is to fit TITL-MARS model, and the second step optimizes TITL-MARS model using MIQP. The benefits of the optimization process has two aspects. First, TITL-MARS can be fit using most commercial MARS software. Second, MIQP can be globally optimized using CPLEX [31].

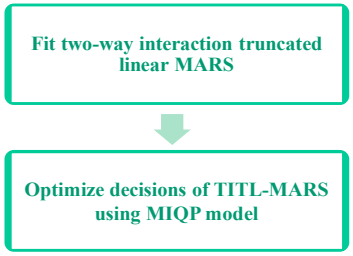


Figure 3.1: TITL-MARS-OPT optimization process

3.5 Experiments and results

In this section, first the genetic algorithm for TITL-MARS optimization is given, and then the presented TITL-MARS-OPT is tested on wind farm power distribution TITL-MARS models and other mathematical models with the genetic algorithm (TITL-MARS-GA) as a benchmark.

3.5.1 Genetic algorithm

The genetic algorithm can also be used as an optimization method to optimize the function (TITL-MARS-GA), as given in Algorithm 3 [32], where the input is the two-way interaction MARS model and the maximum generation number $M_{\max}$, and the output is an optimum and an optimum value. In the "initialization" step (line 1 in Algorithm 3), we generate a population and code the individuals from the decimal form to the binary form. In the "fitness value" step (line 2), we decode the individuals from the binary form to the decimal form and evaluate each of the individual's decimal values in the MARS model function to obtain the fitness value. In the "keep the best" step (line 3), we sort the individuals by their fitness values and store the individual with the best fitness value. The "selection" step (line 6) selects parents from the prior population. The "crossover" (line 7) chooses two parents and produces a new population. The "mutation" (line 8) chooses one point within an individual and changes it from 1 to 0 or from 0 to 1. In this paper, the TITL-MARS-GA algorithm is used as a benchmark compared with the TITL-MARS-OPT method.

Algorithm 3: Genetic algorithm for TITL-MARS optimization

Data: $\hat{f}(\mathbf{x}) = a_0 + \sum_{m=1}^{M} \left\{ a_m \cdot \prod_{k=1}^{Km} [s_{k,m} \cdot (x_{v(k,m)} - t_{v(k,m)})]_+ \right\}, M_{\max}$

Result: $\mathbf{x}_{\max}, f(\mathbf{x}_{\max})$

Initialization: Generate a population and code the individuals from decimal to binary.

Fitness value: Decode individuals from binary to decimal and get function value.

Keep the best: Store the individual with highest or lowest fitness value.

gen = 1

while *gen*< $M_{\max}$ **do**

  Selection: Select parents from prior population.

  Crossover: Choose two parents and produce a new population.

  Mutation: Choose one point and $1 \rightarrow 0$ or $0 \rightarrow 1$.

  Fitness value: Decode individuals from binary to decimal and get function value.

  Keep the best: Store the individual with highest or lowest fitness value.

  gen = gen + 1

end

The parameters of TITL-MARS-GA in this paper are from the literature [32] and given in Table 3.1.

Table 3.1: Parameter settings of TITL-MARS-GA

Parameter	Value
Population size	50
Maximum number of generations	1000
Crossover rate	0.8
Mutation rate	0.15

The TITL-MARS-GA optimization process has two steps as shown in Figure 3.1. The first step fits a two-way interaction truncated linear MARS model, and the second step optimizes decisions of TITL-MARS using the genetic algorithm. The drawback of TITL-MARS-GA is that it does not guarantee global optimality.

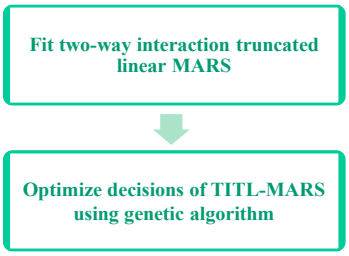


Figure 3.1: TITL-MARS-GA optimization process

3.5.2 Experimental environment

The experiments are run on a workstation with 64 bit Windows 10 Enterprise system. The CPU version is an Intel(R) CPU E3-1285 v6 @ 4.10GHz, and the RAM has 32 GB.The programming code is written in Python version is 3.6, and the CPLEX solver version is 12.8.

3.5.3 Optimization of wind farm power distribution function

Wind farm power is of paramount significance as a renewable energy source. In this paper, the Monte Carlo method [33] is used to generate random wind farm layouts, and the TITL-MARS method is used to study the power distribution under certain wind speeds and directions. After the TITL-MARS model is generated, the TITL-MARS-OPT method is used to study the best turbine position and the worst position. We use the following steps to generate the wind farm power distribution function, as shown in Figure 3.2. First, we randomly generate N wind farm layouts. Second, we calculate average power output at each location. Third, we use the data from second step to build the

TITL-MARS power distribution model.

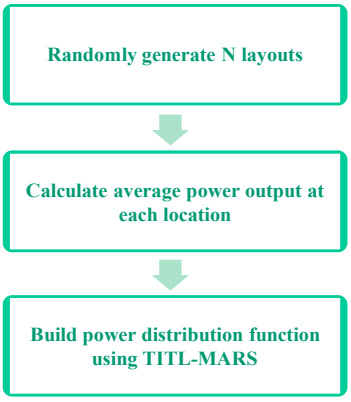
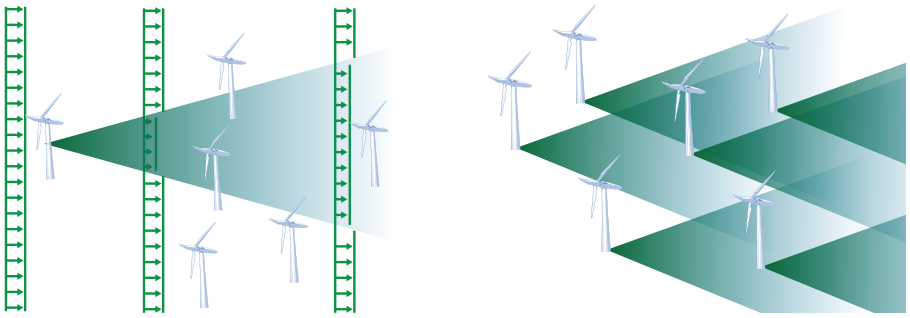


Figure 3.2: Steps to generate a TITL-MARS wind farm power distribution model

After the wind passes through a wind turbine j, a part of the wind energy will be absorbed by turbine j and leave the downstream wind with the reduced speed, which is called the wake effect [34], and the wake effect model is shown in Figure 3.3. Wind speed at turbine i with the wake effect of turbine j is $v_{i,j}$



(a) Wind wake effect model (b) Wake effect illustration

Figure 3.3: Wake effect

and can be calculated as

$$v_{i,j} = v_0 \left(1 - \frac{2}{3} \cdot \frac{R_j^2}{r_j^2}\right). \tag{3.17}$$

R_j is the radius of the wind turbine j, and r_j is the wake radius of the wind turbine j. The final wind speed v_i at turbine i with multiple wake effects is given as

$$v_i = v_0 \left[1 - \sqrt{\sum_{j \in \Phi_i} \left(1 - \frac{v_{i,j}}{v_0}\right)^2}\right], \tag{3.18}$$

where Φ_i is the index set of the turbines which are upwind of the turbine i. Afterwards, the actual power of turbine i can be obtained as [35]

$$p(v_i) = \begin{cases} 0, & v_i < 2 \\ 0.3{v_i}^3, & 2 \leqslant v_i < 12.8 \\ 629.1, & 12.8 \leqslant v_i \leqslant 18 \\ 0, & v_i > 18, \end{cases} \tag{3.19}$$

and the power curve is shown in Figure 3.4. which is the relationship between the wind turbine power and the wind speed.

The wind farm power distribution is generated using the Monte Carlo methods for a given wind farm and a specific wind distribution.

f_{w1} is generated from a wind farm where there is only one wind speed and one direction. The wind farm is divided into 41 by 41 cells, and each cell has a width of 308 m. The wind is from northeast ($\frac{\pi}{4}$) at 15 m/s.

f_{w2} is generated from a wind farm where the wind has only one wind speed and four directions. The wind farm has the same dimension as that of f_{w1}.

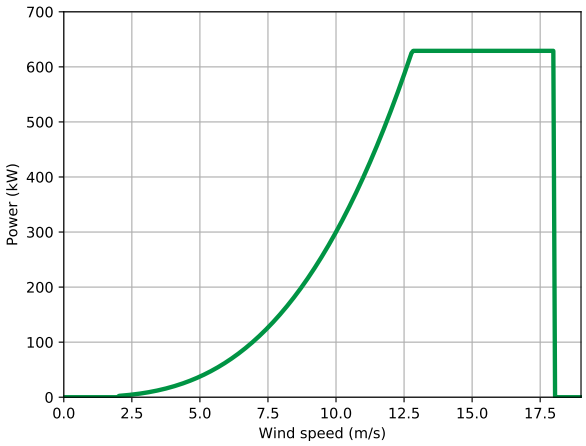


Figure 3.4: Relationship of wind speed with output power of a wind turbine

The wind is from north (0), south (π), east ($\frac{\pi}{2}$), and west ($\frac{3\pi}{2}$) at 15 m/s.

f_{w3} is generated from a wind farm where the wind has only one wind speed at 15 m/s and six directions, 0, $\frac{\pi}{3}$, $\frac{2\pi}{3}$, π, $\frac{4\pi}{3}$, and $\frac{5\pi}{3}$.

f_{w4} is generated from a wind farm where the wind has three wind speeds, 12 m/s, 10 m/s, and 8 m/s, and 12 directions, 0, $\frac{\pi}{6}$, $\frac{\pi}{3}$, $\frac{\pi}{2}$, $\frac{2\pi}{3}$, π, $\frac{7\pi}{6}$, $\frac{4\pi}{3}$, $\frac{3\pi}{2}$, $\frac{5\pi}{3}$, and $\frac{11\pi}{6}$.

TITL-MARS-OPT and TITL-MARS-GA are used to optimize on the wind farm power distribution models to find a global maximum and a minimum, and the results are shown in Figure 3.5 and summarized in Table 3.2. The results are the average value of 30 executions. The table shows the optimal values derived from the TITL-MARS-OPT and TITL-MARS-GA, as well as the computation time in seconds. The result shows that the TITL-MARS-OPT method finds better solutions than the genetic algorithm and uses less time. The maximum value and minimum value are very useful before actually building the wind turbines. The maximum location indicates that it is the

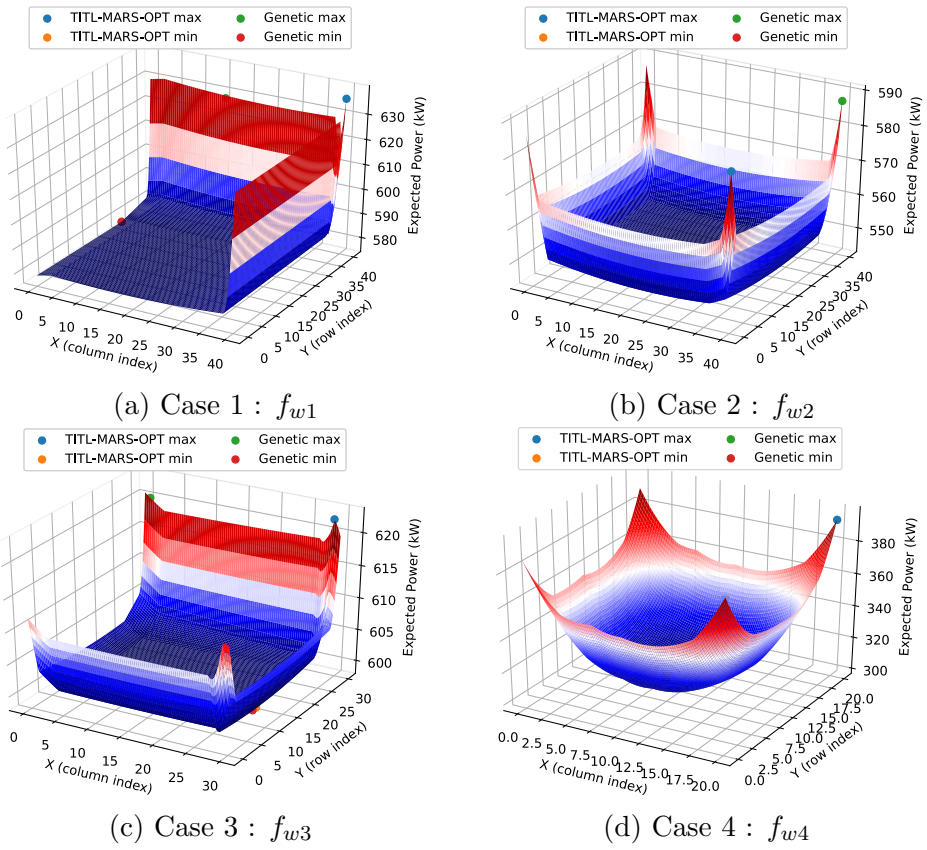
Table 3.2: Comparison of TITL-MARS-OPT and TITL-MARS-GA on wind farm power distribution TITL-MARS models

Function	Measurement	TITL-MARS-OPT	TITL-MARS-GA
f_{w1}	Maximum	636.76	628.12
	Time(seconds)	0.15	5.32
	Minimum	578.30	579.61
	Time(seconds)	0.32	5.46
f_{w2}	Maximum	588.10	586.61
	Time(seconds)	0.15	5.22
	Minimum	545.90	543.81
	Time(seconds)	0.40	5.36
f_{w3}	Maximum	622.10	619.22
	Time(seconds)	0.15	5.18
	Minimum	599.50	601.20
	Time(seconds)	0.85	5.56
f_{w4}	Maximum	393.80	391.46
	Time(seconds)	0.04	2.12
	Minimum	303.21	303.62
	Time(seconds)	0.51	2.26

best location to build a wind turbine based on the given requirements. The minimum location indicates that this location is the worst location on the wind farm, and if the budget is tight, the piece of land around the minimum location can be neglected.

3.5.4 Optimization of other functions

TITL-MARS-OPT method and TITL-MARS-GA are tested to optimize other six TITL-MARS models to find the global maximum and minimum. The first two TITL-MARS models f_1 and f_2 are two-dimensional [36]. The f_3 and f_4 are 10-dimensional TITL-MARS models. f_5 is 19-dimensional, and f_6 is 21-dimensional [37]. The results are shown in Figure 3.6 and summarized in Table 3.3. The results are the average value of 30 runs and show TITL-MARS-



(a) Case 1 : f_{w1}

(b) Case 2 : f_{w2}

(c) Case 3 : f_{w3}

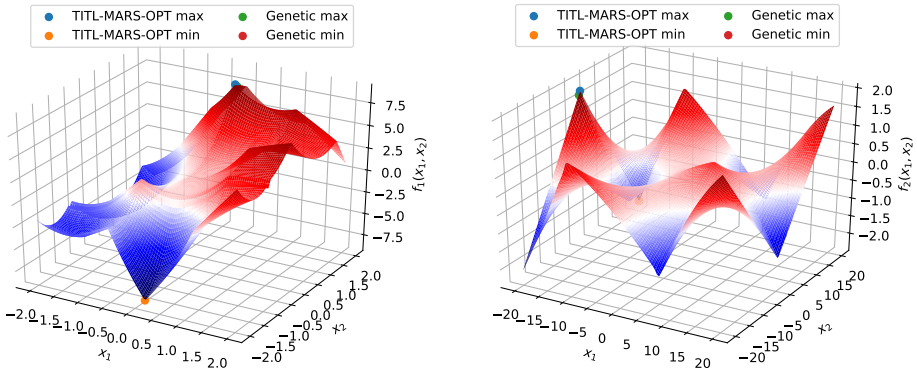
(d) Case 4 : f_{w4}

Figure 3.5: One run result comparison of TITL-MARS-OPT and TITL-MARS-GA on wind farm power distribution TITL-MARS model

Table 3.3: Result comparison of TITL-MARS-OPT and TITL-MARS-GA on six other TITL-MARS mathematical models

Function	Measurement	TITL-MARS-OPT	TITL-MARS-GA
f_1	Maximum	8.30	8.03
	Time(seconds)	0.70	5.42
	Minimum	-8.20	-6.36
	Time(seconds)	1.65	5.56
f_2	Maximum	1.81	1.66
	Time(seconds)	0.31	5.26
	Minimum	-2.20	-2.13
	Time(seconds)	0.32	5.42
f_3	Maximum	5774.08	5532.56
	Time(seconds)	0.02	30.12
	Minimum	-1126.39	-102.21
	Time(seconds)	0.02	30.11
f_4	Maximum	48800.26	10201.13
	Time(seconds)	96.32	34.52
	Minimum	-3952146.24	-3326800.69
	Time(seconds)	0.41	32.27
f_5	Maximum	97679.99	94091.71
	Time(seconds)	0.02	53.61
	Minimum	-15439.62	-6720.36
	Time(seconds)	2.46	53.71
f_6	Maximum	111225.22	100258.76
	Time(seconds)	0.02	54.13
	Minimum	-14215.61	-3102.45
	Time(seconds)	0.04	54.32

OPT is superior to TITL-MARS-GA. The result shows that TITL-MARS-OPT achieves better solutions than TITL-MARS-GA and uses less time, which is consistent with the prior result. The results also show that TITL-MARS-OPT is robust in dealing both low-dimensional and high-dimensional TITL-MARS models.



(a) f_1 (b) f_2

Figure 3.6: One run result comparison of TITL-MARS-OPT and TITL-MARS-GA on f_1 and f_2 TITL-MARS models

3.6 Conclusion

In this paper, a new method (TITL-MARS-OPT) is proposed to globally optimize analytically on the two-way interaction truncated linear MARS (TITL-MARS) by using mixed integer quadratic programming. We verified the presented TITL-MARS-OPT method on the wind farm power distribution TITL-MARS models and six other mathematical TITL-MARS models. The application on wind farm power distribution models gives the best location and worst location information on the wind farm. The testing TITL-MARS models are from 2-dimensions to up to 21-dimensions, and it shows the TITL-MARS-OPT method is robust in dealing with TITL-MARS models with varied dimensions. We also compared the TITL-MARS-OPT method with TITL-MARS-GA in TITL-MARS model optimization, and it shows that the new method can achieve better accuracy and time efficiency. TITL-MARS-OPT can achieve as high as 500% and on average 300% better solution quality and is on average 67% faster than TITL-MARS-GA. In addition, the Python code and the

testing models of this paper are made open source, and it will contribute to the study of TITL-MARS models and optimization.

Bibliography

[1] Chun-Hsin Wu, Jan-Ming Ho, and Der-Tsai Lee. Travel-time prediction with support vector regression. *IEEE transactions on intelligent transportation systems*, 5(4):276–281, 2004.

[2] Yisheng Lv, Yanjie Duan, Wenwen Kang, Zhengxi Li, and Fei-Yue Wang. Traffic flow prediction with big data: a deep learning approach. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):865–873, 2014.

[3] Gregory C Ohlmacher and John C Davis. Using multiple logistic regression and gis technology to predict landslide hazard in northeast kansas, usa. *Engineering geology*, 69(3-4):331–343, 2003.

[4] JR Leathwick, D Rowe, J Richardson, Jane Elith, and T Hastie. Using multivariate adaptive regression splines to predict the distributions of new zealand's freshwater diadromous fish. *Freshwater Biology*, 50(12):2034–2052, 2005.

[5] Jerome H Friedman. Multivariate adaptive regression splines. *The annals of statistics*, pages 1–67, 1991.

[6] Reiner Horst, Panos M Pardalos, and Nguyen Van Thoai. *Introduction to global optimization*. Springer Science & Business Media, 2000.

[7] Lia Morra, Nunzia Coccia, and Tania Cerquitelli. Optimization of computer aided detection systems: An evolutionary approach. *Expert Systems With Applications*, 100:145–156, 2018.

[8] Igor LS Russo, Heder S Bernardino, and Helio JC Barbosa. Knowledge discovery in multiobjective optimization problems in engineering via genetic programming. *Expert Systems with Applications*, 99:93–102, 2018.

[9] Zakaria Abdelmoiz Dahi, Enrique Alba, and Amer Draa. A stop-and-start adaptive cellular genetic algorithm for mobility management of gsm-lte cellular network users. *Expert Systems with Applications*, 106:290–304, 2018.

[10] Mohammed Alswaitti, Mohanad Albughdadi, and Nor Ashidi Mat Isa. Density-based particle swarm optimization algorithm for data clustering. *Expert Systems with Applications*, 91:170–186, 2018.

[11] Ahmed A Ewees, Mohamed Abd Elaziz, and Essam H Houssein. Improved grasshopper optimization algorithm using opposition-based learning. *Expert Systems with Applications*, 112:156–172, 2018.

[12] Adil Baykasoğlu and Fehmi Burcin Ozsoydan. Dynamic optimization in binary search spaces via weighted superposition attraction algorithm. *Expert Systems with Applications*, 96:157–174, 2018.

[13] Christian Bliek1ú, Pierre Bonami, and Andrea Lodi. Solving mixed-integer quadratic programming problems with ibm-cplex: a progress re-

port. In *Proceedings of the twenty-sixth RAMP symposium*, pages 16–17, 2014.

[14] Ozgur Kisi and Kulwinder Singh Parmar. Application of least square support vector machine and multivariate adaptive regression spline models in long term prediction of river water pollution. *Journal of Hydrology*, 534:104–112, 2016.

[15] Sanjiban Sekhar Roy, Reetika Roy, and Valentina E Balas. Estimating heating load in buildings using multivariate adaptive regression splines, extreme learning machine, a hybrid model of mars and elm. *Renewable and Sustainable Energy Reviews*, 82:4256–4268, 2018.

[16] Arpita H Bhatt, Richa V Karanjekar, Said Altouqi, Melanie L Sattler, MD Sahadat Hossain, and Victoria P Chen. Estimating landfill leachate bod and cod based on rainfall, ambient temperature, and waste composition: Exploration of a mars statistical approach. *Environmental Technology & Innovation*, 8:1–16, 2017.

[17] Tanmoy Mukhopadhyay. A multivariate adaptive regression splines based damage identification methodology for web core composite bridges including the effect of noise. *Journal of Sandwich Structures & Materials*, 20 (7):885–903, 2018.

[18] Victoria CP Chen, David Ruppert, and Christine A Shoemaker. Applying experimental design and regression splines to high-dimensional

continuous-state stochastic dynamic programming. *Operations Research*, 47(1):38–53, 1999.

[19] Cristiano Cervellera, Victoria CP Chen, and Aihong Wen. Optimization of a large-scale water reservoir network by stochastic dynamic programming with efficient state space discretization. *European journal of operational research*, 171(3):1139–1151, 2006.

[20] Prashant K Tarun, Victoria CP Chen, HW Corley, and Feng Jiang. Optimizing selection of technologies in a multiple stage, multiple objective wastewater treatment system. *Journal of Multi-Criteria Decision Analysis*, 18(1-2):115–142, 2011.

[21] Julia CC Tsai, Victoria CP Chen, M Bruce Beck, and Jining Chen. Stochastic dynamic programming formulation for a wastewater treatment decision-making framework. *Annals of Operations Research*, 132(1-4): 207–221, 2004.

[22] Julia CC Tsai and Victoria CP Chen. Flexible and robust implementations of multivariate adaptive regression splines within a wastewater treatment stochastic dynamic program. *Quality and Reliability Engineering International*, 21(7):689–699, 2005.

[23] Zehua Yang, Victoria CP Chen, Michael E Chang, Melanie L Sattler, and Aihong Wen. A decision-making framework for ozone pollution control. *Operations Research*, 57(2):484–498, 2009.

[24] Victoria CP Chen, Dirk Günther, and Ellis L Johnson. Solving for an

optimal airline yield management policy via statistical learning. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 52(1):19–30, 2003.

[25] Sheela Siddappa, Dirk Günther, Jay M Rosenberger, and Victoria CP Chen. Refined experimental design and regression splines method for network revenue management. *Journal of Revenue and Pricing Management*, 6(3):188–199, 2007.

[26] Sheela Siddappa, Jay M Rosenberger, and Victoria CP Chen. Optimising airline overbooking using a hybrid gradient approach and statistical modelling. *Journal of Revenue and Pricing Management*, 7(2):207–218, 2008.

[27] Venkata L Pilla, Jay M Rosenberger, Victoria CP Chen, and Barry Smith. A statistical computer experiments approach to airline fleet assignment. *IIE transactions*, 40(5):524–537, 2008.

[28] Venkata L Pilla, Jay M Rosenberger, Victoria Chen, Narakorn Engsuwan, and Sheela Siddappa. A multivariate adaptive regression splines cutting plane approach for solving a two-stage stochastic programming fleet assignment model. *European Journal of Operational Research*, 216(1):162–171, 2012.

[29] Nadia Martinez, Hadis Anahideh, Jay M Rosenberger, Diana Martinez, Victoria CP Chen, and Bo Ping Wang. Global optimization of non-convex

piecewise linear regression splines. *Journal of Global Optimization*, 68(3): 563–586, 2017.

[30] Michael H Kutner, Christopher J Nachtsheim, John Neter, William Li, et al. *Applied linear statistical models*, volume 5. McGraw-Hill Irwin Boston, 2005.

[31] IBM ILOG. Ibm ilog cplex optimization studio cplex users manual, 2018.

[32] Zbigniew Michalewicz. Evolution strategies and other methods. In *Genetic Algorithms+ Data Structures= Evolution Programs*, pages 159–177. Springer, 1996.

[33] Nicholas Metropolis and Stanislaw Ulam. The monte carlo method. *Journal of the American statistical association*, 44(247):335–341, 1949.

[34] Niels Otto Jensen. A note on wind generator interaction. 1983.

[35] Feng Liu and Zhifang Wang. Electric load forecasting using parallel rbf neural network. In *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*, pages 531–534. IEEE, 2013.

[36] Satoshi Miyata and Xiaotong Shen. Free-knot splines and adaptive knot selection. *Journal of the Japan Statistical Society*, 35(2):303–324, 2005.

[37] Bancha Ariyajunya. Adaptive dynamic programming for high-dimensional, multicollinear state spaces. 2013.

Appendix

3.A Supplemental materials

3.A.1 f_1, f_2, f_3 and f_4 functions

$$
\begin{aligned}
f_1(x_1, x_2) =& 3(1-x_1)^2 \exp(-x_1^2 - (x_2+1)^2) - 10(\frac{x_1}{5} - x_1^3 - x_2^5) \exp(-x_1^2 - x_2^2) \\
& - \frac{1}{3} \exp(-(x_1+1)^2 - x_2^2) + 2x_1, \\
& - 2 \leqslant x_1 \leqslant 2, -2 \leqslant x_2 \leqslant 2
\end{aligned}
\tag{3.20}
$$

$$
\begin{aligned}
f_2(x_1, x_2) =& \sin\left(\frac{\pi x_1}{12}\right) \cos\left(\frac{\pi x_2}{16}\right) \\
& - 10 \leqslant x_1 \leqslant 10, -20 \leqslant x_2 \leqslant 20
\end{aligned}
\tag{3.21}
$$

$$f_3(\mathbf{x}) = \sum_{j=1}^{10} \exp(x_j) \left(c_j + x_j - \ln \sum_{k=1}^{10} \exp(x_k) \right) \tag{3.22}$$

$$\mathbf{c} = [-0.6089, -17.164, -34.054, -5.914, -24.721, -14.986, -24.100, -10.708,$$

$$-26.662, -22.179]$$

$$-10 \leqslant x_i \leqslant 10$$

$$f_4(\mathbf{x}) = x_1^2 + x_2^2 + x_1 x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 - 4(x_4 - 5)^2 + (x_5 - 3)^2 \tag{3.23}$$

$$+ 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + 2(x_{10} - 7)^2 + 45$$

$$-10 \leqslant x_i \leqslant 10$$

3.A.2 f_5 and f_6 functions

The datasets to generate f_5 and f_6 are from Ariyajunya [37]. Ariyajunya [37] applied adaptive dynamic programming for high-dimensional, multicollinear state space and used an Atlanta ozone pollution problem as the case study. The datasets used in this paper are from the fourth stage and the third stage with low variance inflation factors.

CHAPTER 4

Conclusion

First, We proposed two new methods for MARS knot positioning, the hill climbing method (HCM) and the hill climbing method using key knots. The HCM and PHCM achieved a reduction in computational time compared to CM, while maintaining similar quality of fit. The PHCM achieved the most significant savings with over 80% reduction in computational time for the higher-dimensional data sets. By using different datasets with different noise levels, we show that PHCM and HCM are robust dealing with noisy datasets.

Second, a new method (TITL-MARS-OPT) is proposed to globally optimize analytically on the two-way interaction truncated linear MARS (TITL-MARS) by using mixed integer quadratic programming. We verified the presented TITL-MARS-OPT method on the wind farm power distribution TITL-MARS models and six other mathematical TITL-MARS models. The application on wind farm power distribution models gives the best location and worst location information on the wind farm. The testing TITL-MARS models are

from 2-dimensions to up to 21-dimensions, and it shows the TITL-MARS-OPT method is robust in dealing with TITL-MARS models with varied dimensions. We also compared the TITL-MARS-OPT method with TITL-MARS-GA in TITL-MARS model optimization, and it shows that the new method can achieve better accuracy and time efficiency. TITL-MARS-OPT can achieve as high as 500% and on average 300% better solution quality and is on average 67% faster than TITL-MARS-GA. In addition, the Python code and the testing models of this paper are made open source, and it will contribute to the study of TITL-MARS models and optimization.