2023-05-01

Generalized Additive Model Using Marginal Integration Estimation Techniques With Interactions

Tahiru Mahama
University of Texas at El Paso

GENERALIZED ADDITIVE MODEL USING MARGINAL INTEGRATION ESTIMATION TECHNIQUES WITH INTERACTIONS

TAHIRU MAHAMA

Master's Program in Mathematical Sciences

APPROVED:

Abhijit Mandal, Ph.D., Chair.

Suneel Babu Chatla, Ph.D.

Sourav Roy, Ph.D.

Stephen Crites, Ph.D.
Dean of the Graduate School

To my

FATHER, Baba Tahiru

with love

GENERALIZED ADDITIVE MODEL USING MARGINAL INTEGRATION ESTIMATION TECHNIQUES WITH INTERACTIONS

by

TAHIRU MAHAMA

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

Department of Mathematical Sciences

THE UNIVERSITY OF TEXAS AT EL PASO

May 2023

Acknowledgement

My special thanks go to the Almighty Lord who has made this journey a success. I also want to extend my profound gratitude to my thesis advisor, Dr. Abhijit Mandal of the Department of Mathematical Sciences at The University of Texas at El Paso for his guidance during this project. I also want to convey my sincere gratitude to Dr. Suneel Babu Chatla from the Department of Mathematical Sciences and Dr. Sourav Roy from the Department of Biological Sciences at The University of Texas at El Paso for their efforts.

Abstract

Marginal Integration (MI) is a statistical method that is extensively employed to estimate component functions of the nonparametric additive models. The shortcoming of the purely additive model is that interaction between predictor variables is often ignored, and it may produce poor performance in some real applications. As a result, this research considers the second-order interactions in the regression models. The primary objective is to use marginal integration techniques to estimate the nonparametric additive functions. We compare this model with other models/estimators such as the Generalized Additive Model (GAM), Generalized Additive Model with Selection (GAMSEL), Robust Marginal Integration (RMI), Ordinary Least Squares (OLS), M-estimators based on Tukey and Huber methods, and LASSO. The simulation results indicate that MI has the least root mean prediction error (RMPE) in pure non-linear models with interaction terms. In the presence of outliers, RMI has the least RMPE demonstrating robustness. Finally, an application of the models on Real Estate Price Prediction data obtained from Kaggle shows that the MI method has the least RMPE depicting as the best model.

Keywords: Marginal Integration, Nonparametric Additive Models, Interaction terms, GAM.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

A linear model uses a statistical model to describe the relationship between an outcome variable and independent variables. These models presuppose a linear relationship between the independent variables and the outcome variable, which means that a change in the independent variables will have a corresponding change in the dependent variable. Simple Linear and Multiple Linear regression are examples of linear models. Simple linear regression is a basic linear model that uses a single predictor variable to explain the variation in a response variable. The simple linear regression model can be expressed as follows:

$$Y = \beta_0 + \beta_1 X + \epsilon,$$

where Y is the response variable, X is the predictor variable, β_0 and β_1 are the intercept and slope coefficients, respectively, and ϵ is the error term, which represents the variability not explained by the model.

Multiple linear regression is a more general linear model that uses multiple independent variables to explain the variation in a response variable. It is represented as

$$Y = \beta_0 + \sum_{j=1}^{d} \beta_j X_j + \epsilon = \beta X + \epsilon,$$

where $X = (1, X_1, X_2, .., X_d)^T$ and $\beta = (\beta_0, \beta_1, \beta_2, .., \beta_d)^T$. An ordinary least squares (OLS) estimator is used to estimate the parameters of the model. However, the OLS often gives a poor estimate in the presence of outliers or other data anomalies in linear models. In these situations, we look for robust estimators which provide accurate results. Tukey's M-estimator and Huber's M-estimator are both robust estimators commonly used in linear

regression to improve the robustness of the model in the presence of outliers or other types of data anomalies. Tukey's M-estimator uses a weighting function to down-weight the influence of outliers on the estimated coefficients. In contrast, Huber's M-estimator combines least squares and absolute deviation approaches to handle both small and large residuals.

In many practical problems with a large number of covariates, one may look for simpler models for better interpretability. So, in those cases, we need an appropriate variable selection method. Variable selection in linear models refers to the process of selecting a subset of predictor variables to include in the model while discarding insignificant variables. One approach to variable selection in linear models is stepwise regression, which adds or removes variables from the model based on statistical criteria. However, it is a very computationally intensive process. Another approach is regularization, which penalizes the coefficients of less important variables to encourage sparsity in the model. Two commonly used types of regularization in linear models are LASSO regression and Ridge regression. However, they are not robust against outliers. The Lad LASSO combines the least absolute deviation (Lad) and LASSO methods, and thus, the estimator produces a robust variable selection.

In statistical modeling, a wide range of nonparametric models falls under the general regression model if a linearity assumption is not tenable, even as a rough approximation. The general regression model is given as

$$Y = m(X) + \sigma(X)\epsilon, \tag{1.1}$$

where

- $X = (X_1, X_2, .., X_d)^T$ is a vector of explanatory variables,
- $m(X)$ and $\sigma(X)$ are functions of X,
- ϵ is independent independent of X with $E(\epsilon) = 0$, $Var(\epsilon) = 1$.

Although nonparametric methods could theoretically be used to estimate the full model (1.1), the curse dimensionality (an unacceptably large increase in variance as the dimension of the data increases) generally makes it difficult. In nonparametric regression models, the dimensionality reduction issue has been addressed by several statisticians. On the other hand, additive models permit the modeling of a response Y as the sum of smooth functions of distinct covariates $X = (X_1, X_2, ..., X_d)^T$. The benefit of additive models over general nonparametric regression models is their ability to avoid the so-called 'curse of dimensionality', caused by the expected number of observations in local neighborhoods declining exponentially as a function of the dimension p of the covariates. The curse of dimensionality is described explicitly by Stone (1985) as "being that the amount of data required to avoid an unacceptably large variance increases rapidly with increasing dimensionality."

Consider m to be additive as a viable middle option for modeling complexity. i.e

$$m(X) = \mu + \sum_{\alpha=1}^{d} f_\alpha(x_\alpha) \tag{1.2}$$

with f_α is unknown and as a main effect.

Leontief (1947) discussed additive models where he looked at the so-called separable functions, which are distinguished by the independence between the changes in the level of one input and the marginal rate of substitution for a pair of inputs. As a result, the additive assumption has been utilized in several areas of economic, statistical, and production theories, such as in connection with the separability hypothesis. In today's theoretical statistics and empirical data analysis, additive models are widely used. They have a desirable statistical structure that makes it possible to conduct statistical analysis on subsets of regressors, decentralize optimization and decision-making, and aggregate inputs into indices (Fuss 1978, Deaton & Muellbauer 1980).

Stone (1985) and Hastie & Tibshirani (1990) suggested additive models, which generalize linear models, resolve the issue of the curse of dimensionality, and provide models

that are easily interpretable as a solution to this problem. The additive components f_α in model (1.2) fulfill some additional need to be recognizable, such as $E(f_\alpha(X_\alpha)) = 0$, to be included in additive models. As in linear regression models, one benefit of additive models is that they enable independent interpretation of each variable's impact on the regression function. In addition, Stone (1985) demonstrated that for these regression models, the best rate for predicting Y is the one-dimensional rate of convergence $n^{-\frac{l}{2l+1}}$, which results in dimensionality reduction through additive modeling. l is the degree of smoothness of f_α.

The asymptotic behavior of the robust estimators of the additive components f_α was introduced by Li et al. (2012) utilizing local linear regression and marginal integration. However, the technique outlined in Li et al. (2012) has the fundamental drawback of only solving the curse of dimensionality when the number of variables, $p \leq 4$, since the local multivariate polynomial under consideration is of order one. This is in addition to presuming that the scale is known. Severance-Lossin & Sperlich (1999) and Kong et al. (2010) described this effect for the traditional estimators based on a local least squares approach, noting that to overcome the problem of dimensionality, the order of the local polynomial approximation should rise with the dimension of the covariates, resulting in higher precision. In an attempt to get solve this issue, Severance-Lossin & Sperlich (1999) changed the initial estimators employed in the integration process, employing higher order kernels and local polynomials that only depend on the covariate X_j associated with the jth additive component to be estimated.

Additive modeling in statistics has been emphasized by some researchers for its usefulness. The requirements of interpretability, dimensionality, and flexibility, which can sometimes be conflicting, are well satisfied by additive models. The curse of dimensionality (an increase in variability as the dimension of the data increases), in particular, can be remedied satisfactorily. Backfitting and splines have typically been used to estimate purely additive models (Hastie & Tibshirani 1990), but marginal integration (Linton & Nielsen 1995) has recently gained some traction due to its ability to construct an explicit

asymptotic theory. Marginal Integration (MI) is a statistical method extensively employed to estimate the average treatment effects of a treatment on an outcome variable. It allows researchers to estimate the effect of a predictor on an outcome variable while holding other predictors constant. Marginal integration is particularly important when dealing with complex models that involve multiple predictors and when there is a need to estimate the effect of a predictor while controlling for the effect of other predictors. One of the key benefits of marginal integration is that it can help to reduce bias in statistical models. By controlling for the effects of other predictors, marginal integration can eliminate the potential confounding effects that can occur when multiple predictors are included in a model. This can lead to more accurate and reliable estimates of the effect of a predictor on an outcome variable.

The shortcoming of the additive model in (1.2) is that interaction between predictor variables is ignored, and the lack of interaction terms has been criticized in several statistical contexts. As a result, this research considers the model's second-order interactions. We use marginal integration techniques to estimate the additive functions of the model. Also, the interaction model will be compared with other models such as the Generalized Additive Model (GAM), Generalized Additive Model with Selection (GAMSEL), Robust Marginal Integration (RMI), Ordinary Least Squares (OLS), Tukey, Huber, and LASSO in terms of their Root Mean Prediction Error. It will help us to compare the advantages and disadvantages of different methods.

Chapter 2

Linear Models

This section talks about existing methods of linear models. The linear models discussed in this section are Ordinary Least Squares (OLS), M-estimators proposed by Tukey and Huber, and LASSO.

2.1 Ordinary Least Squares (OLS)

Linear regression is a frequently used statistical method applied in various fields to study the relationship between an outcome variable and one or more independent variables. It comprises two types; simple linear regression and multiple linear regression. Simple linear regression consists of one predictor variable. Multiple linear regression is an extension of simple linear regression, where there is more than one independent variable. The multiple linear regression model can be represented as:

$$Y_i = \beta_0 + \sum_{i=1}^{d} \beta_k X_{ik} + \epsilon_i, \text{ for } i = 1, 2, ..., n, \tag{2.1}$$

where Y is the dependent variable, X_k's are the independent variables, β_0 is the intercept term, β_k's are the slope coefficients, and ϵ_i is the error term. We assume $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$ and d is the number of covariates in the model.

The above equation can be rewritten in the matrix form as:

$$Y = X\beta + \epsilon,$$

- with $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$,
- Y is an $n \times 1$ vector,
- β is a $(d+1) \times 1$ vector,
- the design matrix X is a $n \times (d+1)$ matrix and
- ϵ is an $n \times 1$ vector.

Each β coefficient represents the change in the dependent variable for a unit change in the corresponding independent variable, holding all other independent variables constant. Multiple linear regression is widely used in various fields, including social sciences, engineering, and business. For example, in social sciences, multiple linear regression is used to study the relationship between various demographic and socio-economic factors and health outcomes. In engineering, multiple linear regression is used to develop predictive models for product design and optimization. In business, multiple linear regression is used to forecast sales and analyze customer behavior.

The regression model requires these four major assumptions:

- Linearity Assumption: $\mu = E(Y|X) = X\beta$
- Independence of the error term: Each ϵ's are independent of each other
- Homoschedasticity: The ϵ's have a constant or equal variance σ^2
- The error term is assumed to be normally distributed.

The coefficients in a multiple linear regression model are estimated using the method of least squares, which involves finding the values of the coefficients that minimize the sum of squared differences between the observed values of the response variable and the predicted values from the model. That is

$$\hat{\beta} = \arg\min_{\beta} Q(\beta),$$

where

$$Q(\beta) = \sum_{i=1}^{n} \left\{ Y_i - \beta_0 - \sum_{j=1}^{d} \beta_j X_{ij} \right\}^2 = (Y - X\beta)^T (Y - X\beta).$$

Differentiating by taking the partial derivative with respect to β gives the gradient

$$\frac{\partial Q(\beta)}{\partial \beta} = -2(X^T Y - X^T X\beta) = -2X^T (Y - X\beta)$$

Setting the partial derivative to zero and solving for β results in

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

and the vector of the fitted values, $\hat{Y}$ is

$$\hat{Y} = X\hat{\beta} = X(X^T X)^{-1} X^T Y = HY$$

with $H = X(X^T X)^{-1} X^T$ termed as the hat matrix/projection matrix.
The advantages of Multiple Linear Regression are

- Flexibility: Multiple linear regression is a flexible and adaptable statistical method that can handle a wide range of research questions and data types.
- Interpretability: The coefficients in multiple linear regression models have a clear interpretation in terms of the relationship between the dependent variable and each independent variable, allowing researchers to make meaningful interpretations of the results.
- Predictive power: Multiple linear regression can be used to make predictions about the dependent variable based on the values of the independent variables, providing researchers with a useful tool for forecasting and decision-making.
- Hypothesis testing: Multiple linear regression allows researchers to test hypotheses

about the relationships between the dependent variable and the independent variables, helping to identify important predictors and explain the variability in the data

Some disadvantages of Ordinary Linear Regression are

- Assumptions: Multiple linear regression relies on a number of assumptions, such as linearity, independence, normality, and constant variance of the errors, which may be violated in practice and can lead to biased or inefficient estimates.
- Multicollinearity: Multiple linear regression can be affected by multicollinearity, where two or more independent variables are highly correlated with each other, making it difficult to separate their effects on the dependent variable.
- Outliers: Multiple linear regression can be sensitive to outliers, which can have a disproportionate influence on the estimated coefficients and lead to biased results.

2.2 M-Estimator

It is used in linear regression to estimate the parameters of the regression line in the presence of outliers. In linear regression, the M-estimator results from optimizing the problem:

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} \rho\left(\frac{y_i - \mathbf{x}_i^\top \boldsymbol{\beta}}{\sigma_{MAD}}\right), \tag{2.2}$$

where

- y_i is the observed value of the response variable,
- $\mathbf{x}_i$ is a vector of predictor variables for the ith observation,
- $\boldsymbol{\beta}$ is a vector of regression coefficients,
- the function ρ can be chosen in such a way to provide the estimator desirable properties in terms of bias and efficiency,

- σ_{MAD} is an estimate of the scale created by the residuals,

$$\sigma_{MAD} = \frac{median|\epsilon_i - median(\epsilon_i)|}{0.6745}.$$

The constant 0.6745 makes the sample mean absolute deviation (MAD) as an unbiased estimator of the population standard deviation σ if n, the sample size is large and normal. Let us define the weight function as follows

$$w_i = \frac{\psi\left(\frac{y_i - \mathbf{x}_i^\top \boldsymbol{\beta}}{\sigma_{MAD}}\right)}{\frac{\left(y_i - \mathbf{x}_i^\top \boldsymbol{\beta}\right)}{\sigma_{MAD}}},$$

where ψ is the first derivative of ρ. It can be shown that the M-estimator of β is obtained by iteratively reweighted least squares (IRLS) algorithm using

$$\hat{\beta} = (X^T W X)^{-1}(X^T W Y),$$

where W is the diagonal matrix with the diagonal element as $w_i,\ i = 1, 2, \cdots, n$.

2.2.1 Huber Estimator

Huber M-estimators were proposed by Peter J. Huber, a Swiss mathematician and statistician, in his paper "Robust Statistics" published in 1964. Considering equation (2.2), the Huber loss function ρ_H is defined as:

$$\rho_H(e) = \begin{cases} \frac{1}{2}e^2 & \text{if } |e| \leq k, \\ k(|e| - \frac{1}{2}k) & \text{if } |e| > k, \end{cases}$$

where $e = y_i - \hat{y}_i$ and k is a tuning constant that controls the transition between the quadratic (least squares) and linear (M-estimator) regimes of the weight function. Taking

the derivative of $\psi(\epsilon)$ gives the weighted function of Huber M-estimator as

$$\psi(e) = \begin{cases} e & \text{if } |e| \leq k, \\ \dfrac{k}{|\epsilon|} & \text{if } |e| > k. \end{cases}$$

The main advantage of Huber M-estimators for linear regression is their robustness to outliers. By giving less weight to observations that deviate from the regression line, Huber M-estimators can provide more accurate estimates of the regression coefficients than least squares regression when the data contains outliers. Moreover, Huber M-estimators are consistent, meaning that as the sample size increases, the estimator converges to the true parameter values. One limitation of Huber M-estimators for linear regression is that they are not as efficient as least squares regression when the data does not contain outliers. In other words, if the data does not contain extreme values, Huber M-estimators may provide less precise estimates than least squares regression. Additionally, the choice of the tuning constant k can be subjective and may require some trial and error to find the best values for a particular dataset.

Overall, Huber M-estimators can be a useful tool in linear regression when dealing with datasets that contain outliers. By providing more accurate estimates of the regression coefficients even in the presence of extreme values, Huber M-estimators can help ensure that the regression line is based on accurate and representative data (Huber 1992, Maronna et al. 2006, Rousseeuw & Leroy 2005).

2.2.2 Tukey's Estimator

Tukey M-estimators can also be used in linear regression models to estimate the regression equation coefficients. They were introduced by John Tukey in 1960 and are particularly useful when the data set contains outliers or other sources of contamination. The Bisquare

function for Tukey estimator is defined as:

$$\rho_T(e) = \begin{cases} \frac{k^2}{6}\left(1 - \left(1 - \left(\frac{e}{k}\right)^2\right)^3\right) & \text{if } |e| \leq k, \\ \frac{k^2}{6} & \text{if } |e| > k. \end{cases}$$

The derivative of $\psi_T(\epsilon)$ gives the weighted function of Tukey estimator as:

$$\psi_T(e) = \begin{cases} e - \left(1 - \left(\frac{e}{k}\right)^2\right)^3 & \text{if } |e| \leq k, \\ 0 & \text{if } |e| > k. \end{cases}$$

The Tukey M-estimator for linear regression has several merits (Tukey 1977, Mosteller & Tukey 1977, Cleveland 1988, Mohd Salleh 2013):

- Robustness: The estimator is resistant to the presence of outliers or other sources of contamination in the data set.
- Efficiency: The Tukey M-estimator for linear regression is efficient when the data set is normal or close to normal.
- Flexibility: The estimator can be easily adapted to different types of distributions by choosing an appropriate tuning constant k.

2.3 Least Absolute Shrinkage and Selection Operator (LASSO)

LASSO penalty, also known as L1 regularization, is a method used to reduce the complexity of linear regression models. In linear regression, the goal is to find the coefficients that best fit the data. However, if the model has too many coefficients, it can overfit the data and result in poor performance on new data. The LASSO penalty addresses this issue by adding

a penalty term to the cost function that encourages sparsity in the coefficient estimates. LASSO estimator of β is obtained by minimizing the problem

$$\min_{||\boldsymbol{\beta}||_1 \leq t} ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2$$

with $||\boldsymbol{\beta}||_1 = \sum_{j=1}^{p} |\beta_j|$ and $t > 0$. Here $\boldsymbol{\beta}$ is a p-dimensional vector of coefficients to be estimated, $\mathbf{y}$ is an n-dimensional vector of observed response variables, $\mathbf{X}$ is an $n \times p$ matrix of observed predictor variables, $||\boldsymbol{\beta}||_1$ is the ℓ_1-norm of $\boldsymbol{\beta}$, defined as the sum of the absolute values of its components, t is a non-negative tuning parameter that controls the amount of shrinkage or sparsity in the estimated coefficients and $||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2$ is the squared Euclidean distance between the observed response $\mathbf{y}$ and the fitted values $\mathbf{X}\boldsymbol{\beta}$.
Applying Lagrangian, the penalized form of the constrained optimization problem is of the form

$$\min_{\boldsymbol{\beta}} ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_1,$$

where $\lambda > 0$ is the tuning parameter that controls the amount of penalization. The LASSO penalty works by shrinking the coefficients of less important predictors to zero, effectively removing them from the model. This helps to simplify the model and reduce the risk of overfitting. However, it can also lead to bias in the coefficient estimates, particularly if there are many correlated predictors.

To choose the optimal value of λ, we can use cross-validation, which involves dividing the data into training and validation sets, fitting the model on the training set, and evaluating its performance on the testing set. This process is repeated for different values of λ, and the value that results in the best performance on the validation set is chosen (Ranstam & Cook 2018, Reid et al. 2016).

Chapter 3

Nonparametric Models

The nonparametric regression models described in this section are Generalized Additive Model (GAM), Generalized Additive Model with Selection, and Robust Partially Linear Additive Model (RPLAM).

3.1 Generalized Additive Model (GAM)

A Generalized Additive Model (GAM) extends the linear regression model, allowing for nonlinear relationships between the explanatory and response variables. The general formula for a GAM can be written as:

$$Y = \mu + \sum_{j=1}^{d} f_j(X_j) + \epsilon,$$

where Y is the outcome variable, $X_1, X_2, ..., X_d$ are the independent variables, μ is the intercept, and ϵ is the error term.

The key difference between a linear regression model and a GAM is the inclusion of the smooth functions $f_1(X_1), f_2(X_2), ..., f_d(X_d)$ in the model. These smooth functions allow for flexible, nonlinear relationships between the independent and dependent variables.

The smooth functions can be represented using techniques such as cubic splines, smoothing splines, or thin-plate splines. The smooth functions used in GAMs can be easily visualized, allowing for easy interpretation of the relationships between variables. The optimal degree of smoothing is chosen based on the data and the desired trade-off between flexibility and simplicity.

GAMs can also incorporate interaction terms between the independent variables to allow for nonlinear interactions between the variables. The general formula for a GAM with interactions can be written as:

$$Y = \mu + \sum_{j=1}^{d} f_j(X_j) + \sum_{1 \leq j < k \leq d} f_{jk}(X_j, X_k) + \epsilon,$$

where $f_{jk}(X_j, X_k)$ is the smooth function for the interaction between the jth and kth independent variables.

The key advantage of GAMs is that they allow for flexible modeling of nonlinear relationships between predictors and the response variable without requiring prior knowledge or assumptions about the functional form of these relationships. This makes them very useful when traditional linear regression models do not capture the true relationship between the independent variables and the response variable. GAMs can be used in various applications, including environmental science, finance, social sciences, and medical research. They are particularly useful when working with large datasets, where it is difficult to identify the exact form of the relationship between the predictors and the response variable (Hastie & Tibshirani 1990)

The basic structure of a GAM is similar to that of a linear regression model, with a response variable and a set of independent variables. However, in GAMs, the predictor variables are modeled as smooth functions, often represented by spline functions, allowing for flexible modeling of nonlinear relationships. The smooth functions are estimated using maximum likelihood or Bayesian methods. One of the key features of GAMs is that they can handle multiple predictors simultaneously and model their interactions. This allows for the identification of complex relationships between variables, which may not be evident in a linear regression model (Mammen et al. 1999)

GAMs can be fitted using maximum likelihood estimation, and the optimal smoothing

parameters can be chosen using various methods, such as cross-validation or information criteria. The models can be fitted using a variety of software packages, such as R's "mgcv" package. generalized additive models (GAMs) have some disadvantages, such as being computationally intensive and requiring larger sample sizes than linear regression models. Additionally, GAMs can overfit the data if too many variables or too much flexibility is included in the model. The choice of smoothing parameters in GAMs is subjective and may impact the model's results. These requirements should be taken into account when using GAMs for statistical modeling.

3.2 Generalized Additive Models with Selection (GAMSEL)

Nonparametric Generalized Additive Models with Selection (GAMSEL) is a statistical model used for regression analysis. They have been well known in recent years due to their ability to handle complex, high-dimensional data and identify significant variables (Hastie & Tibshirani 1990). Nonparametric GAMSEL is a variant of generalized additive models (GAMs), flexible regression models that allow for nonlinear relationships between variables. GAMSELs extend the GAMs by incorporating variable selection into the model fitting process. This selection process can help identify essential variables, reduce overfitting, and improve model interpretability. Nonparametric GAMSEL further extends GAMSEL by using nonparametric smoothing techniques to model the functional relationship between variables. The objective function of GAMSEL is defined as:

$$\hat{f}_1, \ldots, \hat{f}_p = \arg \min_{f_1,\ldots,f_p \in F} L(Y; f_1, \ldots, f_p) + \sum_{j=1}^{p} J(f_j),$$

where

- $L(Y; f_1, \ldots, f_p)$ is the negative log-likelihood of the data, given the predictor variables

X and the response variable Y

- $f_1, \ldots, f_p$ are smooth functions of the predictor variables, which are modeled using spline functions or other basis functions
- $J(f_j)$ is a penalty term applied to the j-th smooth function f_j, which can be expressed as a function of the type of basis functions used and the degree of smoothness. The penalty term $J(f_j)$ penalizes the complexity of the smooth function f_j by constraining the roughness penalty of the function. This encourages the function to be smooth and prevents overfitting.

The form of the penalty term $J(f_j)$ depends on the type of basis functions used to model the smooth function f_j and the degree of smoothness. In general, $J(f_j)$ penalizes deviations from a simple or smooth function, while allowing some flexibility in the functional form. For example, if the smooth function f_j is modeled using cubic spline basis functions, the penalty term $J(f_j)$ can be expressed as:

$$J(f_j) = \lambda_j \int_{\mathbb{R}} \left[f_j''(x)\right]^2 dx,$$

where $f_j''(x)$ is the second derivative of the function f_j with respect to x, and λ_j is the penalty parameter that controls the amount of regularization applied to f_j. This penalty term encourages the smoothness of f_j by penalizing large values of its second derivative.

The GAMSEL method optimizes the objective function by using a backfitting algorithm, which alternates between updating each smooth function f_j while keeping the others fixed, and updating the penalty parameter λ_j for each smooth function f_j while keeping the others fixed. This iterative process continues until convergence is reached. The regularization parameter λ introduced in GAMSEL is to regulate the amount of shrinkage applied to the smooth functions, and the model is estimated by minimizing the sum of squared residuals subject to the condition that the sum of the absolute values of the coefficients is less than or equal to λ. One of the key advantages of nonparametric GAMSEL is its ability to handle

complex, high-dimensional data. In many real-world applications, data sets can be large, noisy, and contain many variables, making it difficult to fit a traditional linear model. Nonparametric GAMSEL allows for a flexible modeling approach that can capture complex nonlinear relationships between variables, while also identifying the most important variables in the data set.

However, there are also some limitations to nonparametric GAMSEL. One limitation is that the model fitting process can be computationally intensive, especially for large data sets or when using high-dimensional smoothing techniques. Additionally, the interpretation of the model can be more challenging than with linear models, as the functional relationships between variables can be complex and difficult to explain (Chen et al. 1996, Green & Silverman 1993).

3.3 Robust Partially Linear Additive Model (RPLAM)

The Robust Partially Linear Additive Model (RPLAM) is a statistical model that extends the Generalized Additive Model (GAM) by including both linear and nonlinear predictor variables, while also being robust to the presence of outliers in the data. RPLAM is particularly useful when dealing with complex datasets where the relationships between variables are nonlinear, and the presence of outliers may distort the results. The RPLAM consists of a linear model and a nonparametric component. The linear component involves a set of linear predictor variables, while the nonparametric component involves a set of nonlinear predictor variables that are represented as smooth functions. The smooth functions are typically represented using spline functions, and the number of knots is chosen using cross-validation or a similar method.

Robust partially Linear Additive Model is also a type of GAM model that uses a penalized likelihood approach to estimate regression parameters. The regularization parameter is estimated using cross-validation. The robustness of the RPLAM is achieved through the

use of a robust estimator of scale, such as the Huber estimator, which down weights the effect of outliers in the data. The robust estimator of scale is combined with a robust estimation procedure for the nonlinear component of the model, which ensures that the model is robust to the presence of outliers. RPLAM has a wide range of applications, including finance, economics, and medical research. In finance, for example, RPLAM can be used to model the relationship between stock prices and various economic indicators, while also taking into account the presence of outliers that may be caused by market shocks or other unexpected events.

The RPLAM model can be formulated as:

$$Y = X^T\beta + g(Z) + \epsilon, \tag{3.1}$$

In this model

- Y is an n $\times$ 1 response vector
- X is an n $\times$ p predictor variables, with p being the number of predictors
- β is a p $\times$ 1 vector of regression coefficients for the linear component of the model
- $g(Z)$ is a nonlinear function of a set of covariates Z, estimated using penalized regression splines
- ϵ is n $\times$ 1 vector of error terms

Jiang (2015) proposed a robust estimator for RPLAM where he introduced the Least Squares Method. Let $(Y_i, X_i^T, Z_i)^T$, $i = 1, ..., n$ be the random samples that satisfies the partially linear additive model in eqn 2.1. Let us define

$$\phi_0(Z_i) = E(Y|Z = z),$$

$$\phi(Z) = E(X|Z = z).$$

Then

$$g(z) = \phi_0(Z) - \phi_0(z)^T\beta.$$

Hence, equation (3.1) becomes

$$Y_i - \phi_0(Z_i) = (X_i - \phi(Z_i))^T\beta + \epsilon_i, \qquad i = 1, ..., n \tag{3.2}$$

Since $\phi_0(\cdot)$ and $\phi(\cdot)$ are unknown, we can find their suitable estimates by letting

$$W_{nj}(z) = \frac{K\left(\frac{z-Z_j}{h_n}\right)}{\sum_{j=1}^n K\left(\frac{z-Z_j}{h_n}\right)},$$

where $K(\cdot)$ is a kernel function and h_n is a bandwidth tending to zero. The classical nonparametric method estimates the conditional expectations through

$$\hat{\phi}_0(z) = \sum_{j=1}^n W_{nj}(z)Y_j, \qquad \hat{\phi}(z) = \sum_{j=1}^n W_{nj}(z)X_j. \tag{3.3}$$

Hence, the LS estimator $\hat{\beta}_n$ of β can be derived by minimizing

$$\sum_{i=1}^n (Y_i - \hat{\phi}_0(Z_i) - (X_i - \hat{\phi}(Z_i))^T\beta)^2,$$

and the LS estimate of unknown smooth function g is given as

$$\hat{g}(z) = \hat{\phi}_0(z) - \hat{\phi}(z)^T\hat{\beta}_n.$$

The procedure to obtain robust estimators is as follows. We first estimate $\phi_0(\cdot)$ and $\phi(\cdot)$ by a robust smoothing using a local median estimate proposed by Bianco et al. (2011). The resulting robust estimators are denoted as $\tilde{\phi}_0(z)$ and $\tilde{\phi}(z)$, respectively. Let $Y_i^* = Y_i - \tilde{\phi}_0(Z_i)$ and $X_i^* = X_i - \tilde{\phi}_0(Z_i)$. An affine equivariant robust estimator of the unknown parameter

β can be obtained by maximizing

$$\widetilde{\beta}_n = \arg\max_{\beta \in \mathbb{R}^d} \sum_{i=1}^{n} \exp\left\{ -\frac{(Y_i^* - X_i^{*T}\beta)^2}{\hat{\sigma}\lambda} \right\},$$

where $\lambda \in (0, \infty)$ (is a tuning parameter which controls the degree of robustness and efficiency), $\hat{\sigma}$ is a median absolute deviation (MAD) estimator based on the residuals $Y_i^* - X_i^{*T}\widetilde{\beta}_{INI}, i = 1, \ldots, n$ with an initial estimate $\widetilde{\beta}_{INI}$ of β.
Finally, we define the estimate of unknown smooth function g as $\widetilde{g}(z) = \widetilde{\varphi}_0(z) - \widetilde{\varphi}(z)^T\widetilde{\beta}_n$.

Although the Robust Partially Linear Additive Model (RPLAM) has several benefits, such as the ability to model both linear and nonlinear predictor variables and robustness to the presence of outliers, there are also some disadvantages to consider. These include:
Complexity: RPLAM is a complex model that estimates linear and nonlinear components. This can make the model difficult to interpret and may require advanced statistical knowledge to use effectively. Computational burden: The estimation of RPLAM requires the estimation of both linear and nonlinear components, which can be computationally intensive, especially when dealing with large datasets.
Assumptions: RPLAM relies on certain assumptions, such as linearity and additivity, like any statistical model. When these assumptions are violated, estimates may be biased or inefficient.
Limited applicability: RPLAM may not be suitable for all types of data, especially when dealing with highly complex relationships or sparse data. In such cases, alternative modeling approaches may be more appropriate. Overall, while RPLAM is a powerful and flexible model, it is not without its limitations. Researchers should carefully consider these limitations and evaluate alternative approaches before deciding to use RPLAM (Sun & Liu 2022)

Chapter 4

GAM with Interaction Effect

One of the methods for this study is generalized additive model (GAM) with interaction terms using the marginal integration estimation. In many additive models, interaction terms are completely ignored, this method seeks to address the issue of interaction terms. Bivariate interaction terms for the marginal integration estimator have been investigated by Sperlich et al. (2002). They provide asymptotic properties and also introduce test methods to determine the relevance of the interactions.

4.1 Marginal Integration

We define the nonparametric GAM with interactions as:

$$Y = c + \sum_{\alpha=1}^{d} f_\alpha(X_\alpha) + \sum_{1\leq\alpha<\beta\leq d} f_{\alpha\beta}(X_\alpha, X_\beta) + \sigma(X)\epsilon, \tag{4.1}$$

where

- $X = (X_1, X_2, ..., X_d)$ is a sequence of independent and identically distributed (i.i.d) vectors of explanatory variables,
- ϵ is also a sequence of iid random variables independent of X such that $E(\epsilon) = 0$ and $Var(\epsilon) = 1$,
- $[f_\alpha(.)]_{\alpha=1}^{d}$ and $[f_{\alpha\beta}(.)]_{1\leq\alpha<\beta\leq d}$ are real-valued unknown functions.

According to Linton & Nielsen (1995) and Tjøstheim & Auestad (1994), the marginal influence of X_α, X_β and (X_α, X_β) are estimated by the integration estimators as follows.

Main Effect:

$$\hat{F}_\alpha(X_\alpha) = \frac{1}{n}\sum_{l=1}^{n} \hat{m}_\alpha(X_\alpha, X_\beta, X_{l\alpha}).$$

Interaction Effect:

$$\hat{F}_{\alpha\beta}(X_\alpha, X_\beta) = \frac{1}{n}\sum_{l=1}^{n} \hat{m}_{\alpha\beta}(X_\alpha, X_\beta, X_{l\alpha\beta}),$$

where $X_{l\alpha\beta}(X_{l\alpha})$ is the lth observation of X with X_α and $X_\beta(X_\beta)$ removed.
$\hat{m}_\alpha(X_\alpha, X_\beta, X_{l\alpha\beta})$ is the local preestimator which is computed using multidimensional local kernel estimation.

For the pre-estimator of the main effect:

$\hat{m}_\alpha(X_\alpha, X_{l\alpha'}) = e_1(Z_\alpha^T W_{1,\alpha} Z_\alpha)^{-1} Z_\alpha^T W_{1,\alpha} Y$ where each of the components are defined as:

$$e_1 = (1, 0)$$

$$W_{1,\alpha} = diag\left[\frac{1}{n} K_h(X_{i\alpha} - x_\alpha) L_g(X_{i\alpha'} - X_{l\alpha'})\right]_1^n$$

$$Z_\alpha = \begin{bmatrix} 1 & X_{1\alpha} - x_\alpha \\ . & . \\ . & . \\ . & . \\ 1 & X_{n\alpha} - x_\alpha \end{bmatrix}.$$

For the preestimator of the interaction effect:

$$\hat{m}_{\alpha\beta}(X_\alpha, X_\beta, X_{l\alpha'\beta'}) = e_1(Z_{\alpha\beta}^T W_{1,\alpha\beta} Z_{\alpha\beta})^{-1} Z_{\alpha\beta}^T W_{1,\alpha\beta} Y,$$

where each of the components are defined as:
$Y = (Y_1, Y_2, ..., Y_n)^T$ is an n by 1 vector

$$W_{1,\alpha\beta} = diag\left[\frac{1}{n} K_h(X_{i\alpha} - x_\alpha, X_{i\beta} - x_\beta) L_g(X_{i\alpha'\beta'} - x_{l\alpha'\beta'})\right]_1^n$$

K and L are two dimensional kernel functions and h is the bandwidth
$K_h(.) = h^{-1} K\left(\frac{.}{h}\right), \quad L_g(.) = g^{-1} K\left(\frac{.}{g}\right) \quad e_1 = (1, 0, 0)$

$$Z_{\alpha\beta} = \begin{bmatrix} 1 & X_{1\alpha} - x_\alpha & X_{1\beta} - x_\beta \\ . & . & . \\ . & . & . \\ . & . & . \\ 1 & X_{n\alpha} - x_\alpha & X_{n\beta} - x_\beta \end{bmatrix}.$$

4.2 Robust Marginal Integration (RMI)

Robust marginal integration (RMI) is among the statistical techniques that can enhance the performance of generalized additive models (GAMs) when dealing with non-normal or non-Gaussian data. The method combines the advantages of both marginal integration and robust estimation techniques to create a more robust and accurate model. Considering the additive model,

$$Y = \mu + \sum_{\alpha=1}^{d} g_\alpha(X_\alpha) + \sigma(X)\epsilon$$

where the error term ϵ is independent of X and has a symmetric distribution F_0. The robust estimator was proposed by Boente et al. (2017).

The robust estimator for the α-th component of the additive model is obtained through a marginal integration procedure. Essentially, this involves integrating out all the other components of the model and only considering the function of interest. The resulting estimator is a weighted average of the initial local polynomial M-estimators, where the weights are given by the density function of the residuals of the M-estimator.

Let $X = (X_1, \ldots, X_d)$ be the vector of covariates, and let $g_\alpha(x_\alpha)$ be the marginal regression function of interest for the α-th covariate. Let q be the order of the initial local polynomial M-estimator used to estimate each $g_\alpha(x_\alpha)$ is defined as:

$$\hat{g}(x) = \frac{1}{d} \sum_{\alpha=1}^{d} g_{\alpha,mq,\alpha}(x_\alpha),$$

where $g_{\alpha,mq,\alpha}(x_\alpha)$ is the mq-th order local polynomial M-estimator of $g_\alpha(x_\alpha)$ based only on the α-th component of X and can be estimated using a weighted least squares method.

The RMI process can be implemented using various software packages that support generalized additive models, such as the 'gam' function in the R package 'mgcv'. The 'gam' function allows users to specify the type of robust estimator to use in the RMI process, as well as other parameters such as the choice of basis functions for the smooth terms and the type of penalization for the smoothness parameters. Other packages such as 'rmargint' proposed by Boente et al. (2017) can implement RMI in R.

Chapter 5

Simulation Results

We perform a simulation experiment to validate the performance of the models based on their Root Mean Prediction Error (RMPE). 3 Predictors (X_1, X_2, X_3) are simulated from a uniform distribution with sample sizes of n = 50, 100, 200. Onwards, estimating the main effects and interaction effects, we assumed the following functions:

$f_1 = 24(X_1 - 0.5)^2 - 2$

$f_2 = 2\pi \sin(\pi X_2) - 4$

These functions (f_1, f_2) are nonlinear functions used in the response variable (Y).

$$Y = f_1(X_1) + f_1(X_2) + X_3 + \log(1 + X_1 X_2) + \exp(X_2 X_3) + \epsilon,$$

where

$$\epsilon \sim (1-p)N(0, \sigma^2) + pN(20, \sigma^2).$$

where p represents the proportion of outliers and we have taken $p = 0, 0.05$ and $\sigma = 1$. Also, we have generated full training data, 5% trimmed training data, and test data with the same predictors and then applied RMI (Robust Marginal Integration), MI (Marginal Integration), OLS (Ordinary Least Squares), Huber regression, Tukey regression, LASSO (Least Absolute Shrinkage and Selection Operator), GAM (Generalized Additive Model) and GAMSEL (Generalized Additive Model with Selective Estimation of Parameters) for prediction. The simulation is replicated 100 times, and each model Root Mean Prediction Error (RMPE) is computed for the full training data set, 5% trimmed training data, and testing data. Since the MI and RMI method require the selection of bandwidth parameters, we have tuned the bandwidths for different sample sizes. The R code used for this simulation

can be found in the Appendix. The bandwidth parameters used for this simulation are also mentioned at the end of the code.

5.1 Main Effect and Interaction Plots

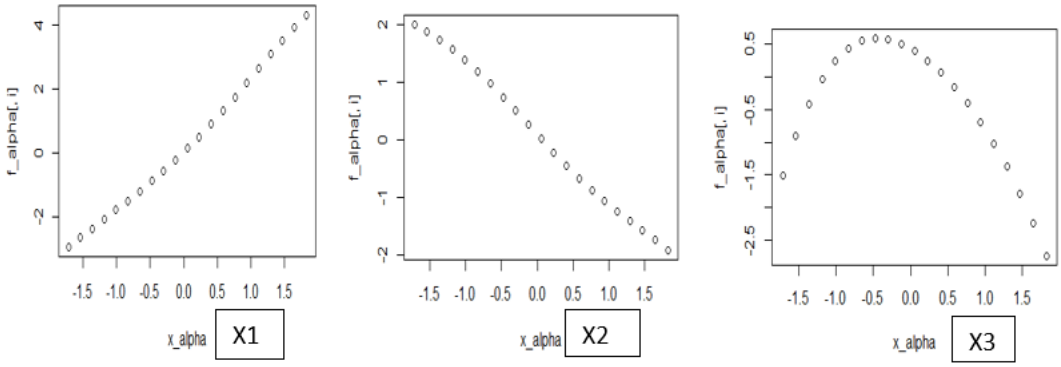

Figure 5.1: Estimates of the main effect using the Marginal Integration method.

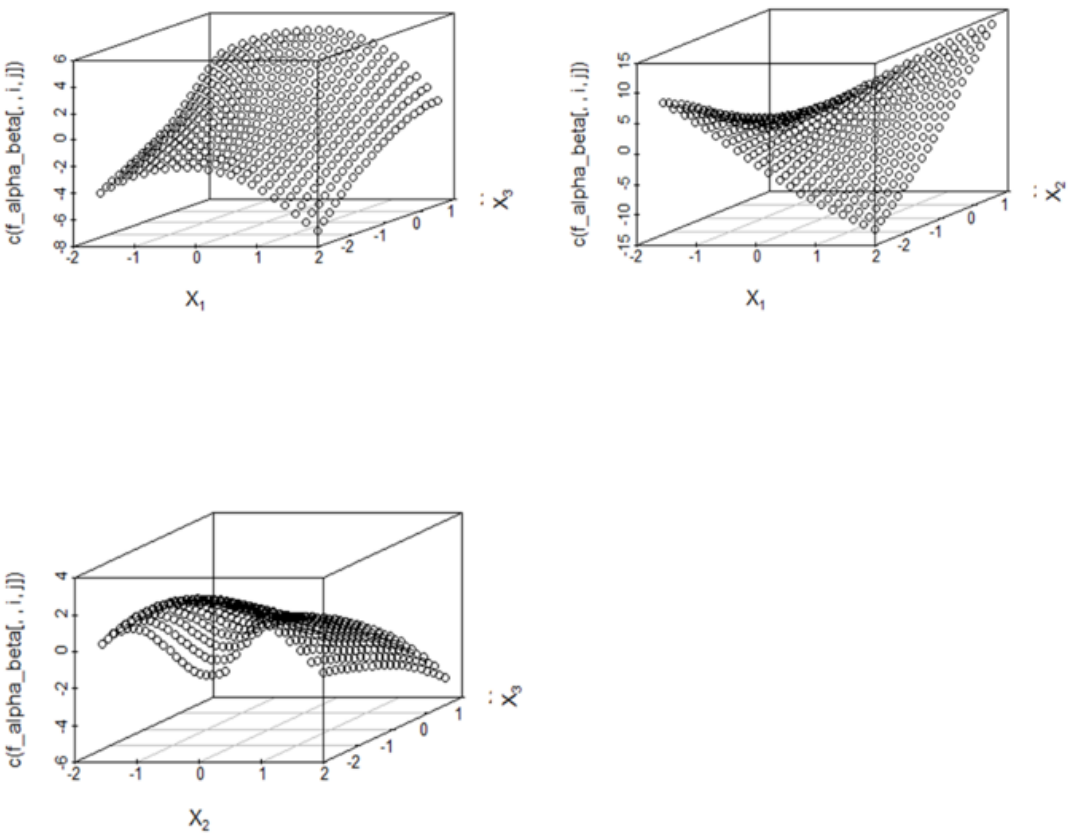

Figure 5.2: Estimates of the interaction effect using the Marginal Integration method.

5.2 Root Mean Prediction Error (RMPE)

Root Mean Prediction Error (RMPE) is a measure of the average deviation between the predicted values and actual values of model. The RMPE is defined as

$$\text{RMPE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2},$$

where n is the number of observations, y_i is the actual value of the i-th observation, and $\hat{y}_i$ is the predicted value of the i-th observation.

5.3 RMPE for Simulated Data

In this section, we have reported the Root Mean Prediction Error (RMPE) for the full training data set, 5% trimmed training data, and testing data using n = 50, 100 and 200 for different proportions of outliers (p = 0%, 5%). The main idea is to help check the nature of the RMPE of the models when the proportion of outliers is altered.
Table 4.1 depicts the RMPE for $n = 50$, $p = 0\%$. The RMPE for the nonlinear models (MI, RMI, GAM and GAMSEL) appear to be relatively minimal as compared to the RMPE for linear models (OLS, Tukey, Huber and LASSO) in all three cases for full training data, 5% trimmed training data and test data. The MI method has the least RMPE for the test data, followed by the RMI method and the other linear models.

Table 5.1: RMPE of different estimators for n = 50, p = 0.

Data / Estimator	Full Training Data	5% Trimmed Training Data	Test Data
MI	1.4956194	1.1343427	1.693024
RMI	1.9509687	1.7033965	1.880585
GAM	1.8341465	1.7184733	1.909057
GAMSEL	1.8792736	1.7567732	1.904757
OLS	2.4231850	2.1378903	2.925841
Tukey	2.4274825	2.1236149	2.945889
Huber	2.4277466	2.1230090	2.947578
LASSO	2.4500079	2.1606034	2.929164

Similarly, Table 4.2 shows the RMPE for 5% contaminated proportion of outliers. It can be seen that almost the nonlinear models (RMI, MI, GAM, and GAMSEl) have relatively minimal RMPE than the RMPE of the linear models (OLS, Tukey, Huber, and LASSO), with RMI being more robust, followed by MI model. RMI down weights outliers within an observation leading to relatively minimal RMPE.

Table 5.2: RMPE of different estimators for n = 50, p = 0.05.

Data / Estimator	Full Training Data	5% Trimmed Training Data	Test Data
MI	3.334617	2.150334	2.244626
RMI	3.341925	1.866634	2.063283
GAM	3.415981	1.641068	2.367746
GAMSEL	3.744582	1.588817	2.087379
OLS	4.481427	2.547037	3.381487
Tukey	4.650484	2.298392	2.946648
Huber	4.577265	2.314059	3.021648
LASSO	4.578127	2.572043	3.312549

Table 5.3: RMPE of different estimators for n = 100, p = 0.

Data / Estimator	Full Training Data	5% Trimmed Training Data	Test Data
MI	1.0883718	1.0371954	1.0932077
RMI	1.1819926	1.3872637	1.2025745
GAM	0.9198588	0.8012661	1.120111
GAMSEL	0.9377968	0.8168821	1.100418
OLS	2.7307057	2.4398454	2.521105
Tukey	2.7333633	2.4312190	2.532443
Huber	2.7345418	2.4304282	2.534311
LASSO	2.7649038	2.4649656	2.500821

Table 5.4: RMPE of different estimators for n = 100, p = 0.05.

Estimator \ Data	Full Training Data	5% Trimmed Training Data	Test Data
MI	4.244543	1.537812	1.317160
RMI	4.131838	1.291164	1.141759
GAM	4.157817	1.822049	2.153853
GAMSEL	4.303037	1.710540	1.833162
OLS	5.130159	3.002963	2.908784
Tukey	5.278511	2.732754	2.530476
Huber	5.209252	2.759724	2.580387
LASSO	5.180242	2.990368	2.802996

Table 5.5: RMPE of different estimators for n = 200, p = 0.

Estimator \ Data	Full Training Data	5% Trimmed Training Data	Test Data
MI	1.3783011	1.9248289	1.0351418
RMI	1.5367982	1.3550877	1.4029131
GAM	0.9769661	0.8529152	0.9692405
GAMSEL	0.9855903	0.8605986	0.9627929
OLS	2.7759304	2.4773668	2.7182039
Tukey	2.7767363	2.4741015	2.7267952
Huber	2.7770788	2.4734560	2.7303227
LASSO	2.7823566	2.4857525	2.6971132

Table 5.6: RMPE of different estimators for n = 200, p = 0.05.

Estimator \ Data	Full Training Data	5% Trimmed Training Data	Test Data
MI	4.651391	1.752412	1.644437
RMI	4.664832	1.565018	1.382834
GAM	4.329318	1.653548	1.626122
GAMSEL	4.399191	1.603645	1.471823
OLS	5.179609	3.053820	2.962892
Tukey	5.261134	2.768808	2.726325
Huber	5.206081	2.784423	2.698031
LASSO	5.155451	3.005571	2.741744

Finally, in the simulation results on the RMPE in Table 4.3, 4.4, 4.5 and 4.6, RMI has the least RMPE indicating robustness to outliers where there is 5% proportion of outliers. Also, MI has the least RMPE when there is 0% proportion of outliers. In most cases, the nonlinear models have relatively minimal RMPE than the linear model. The RMI proposed by Boente et al. (2017) down-weights outliers resulting in relatively minimal RMPE.

Chapter 6

Real Data Analysis

A dataset on Real Estate Price Prediction is obtained from Kaggle (*Kaggle Real Estate Price Prediction Dataset* n.d.) which has 414 observations across 7 variables, of which one is a response variable, and 6 are predictors. The house price is the response variable, and among the predictor variables are house age, distance to the nearest MRT station, number of convenience stores, latitude, longitude, and transaction date. All variables are continuous except the transaction date. Since the Marginal Integration method works for only continuous variables, the analysis will discard the transaction date.

We fit a GAM model with interactions using Marginal Integration Estimation Techniques. It is applied to Real Estate Price data where the response variable is house price, and the response variables are various features of house age, distance to the nearest MRT station, number of convenience stores, latitude, and longitude. The method involves estimating the main and interaction effects using a kernel function and integrating over the covariates space to obtain the estimated function. The main effect for each covariate is estimated, where the output GAM model is a list containing the estimated and interaction effects for each covariate. The GAM function first scales the predictors in the predictor matrix before estimating the main effects of each predictor using the main effect function, which predicts the marginal effects of each predictor using the Gaussian kernel density estimator with bandwidth. Nevertheless, the GAM function also calculates the interaction effect between each pair of predictors using the interaction effect function, which computes the interaction effect at each pair of points using the product kernel density estimator. Onwards, to check the model's performance, we developed a prediction function that predicts the response variable based on the estimated main and interaction effects. The dataset is

split into 75% training dataset and 25% testing dataset. The model is built on the training dataset and later validated on the testing dataset based on the Root Mean Prediction Error (RMPE). Furthermore, the RMPE is extended to the Generalized Additive Model (GAM), GAMSEL, RMI, OLS, Tukey, Huber, and LASSO.

6.1 Main Effect and Interaction Plots

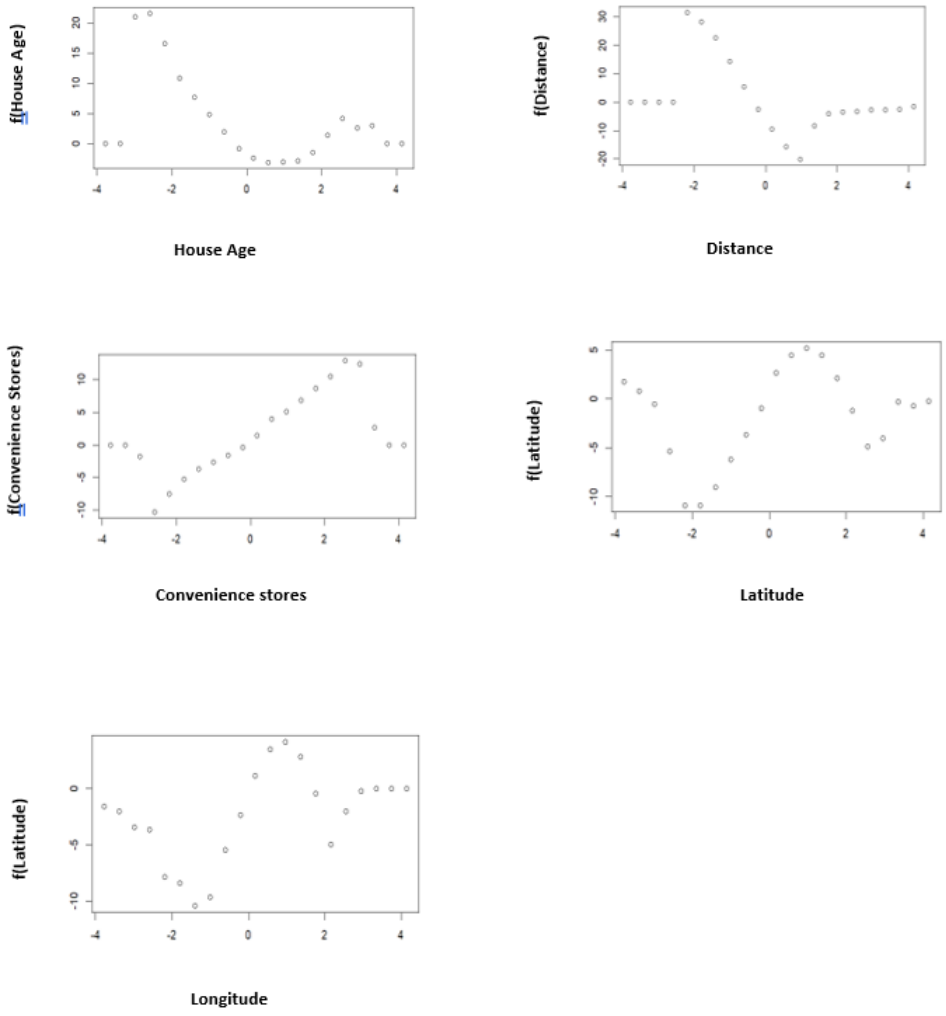

Figure 6.1: Main Effect Plots

Figure 4.3 depicts the plots of the main effects (the plot of the main effect against each predictor). One requirement of the additive model is that the relationship between the response variable and predictors must be nonlinear. The plot above depicts a nonlinear relationship between the response variable (House Prices) and all other predictors. These variables will be suitable for fitting a generalized additive nonparametric model.

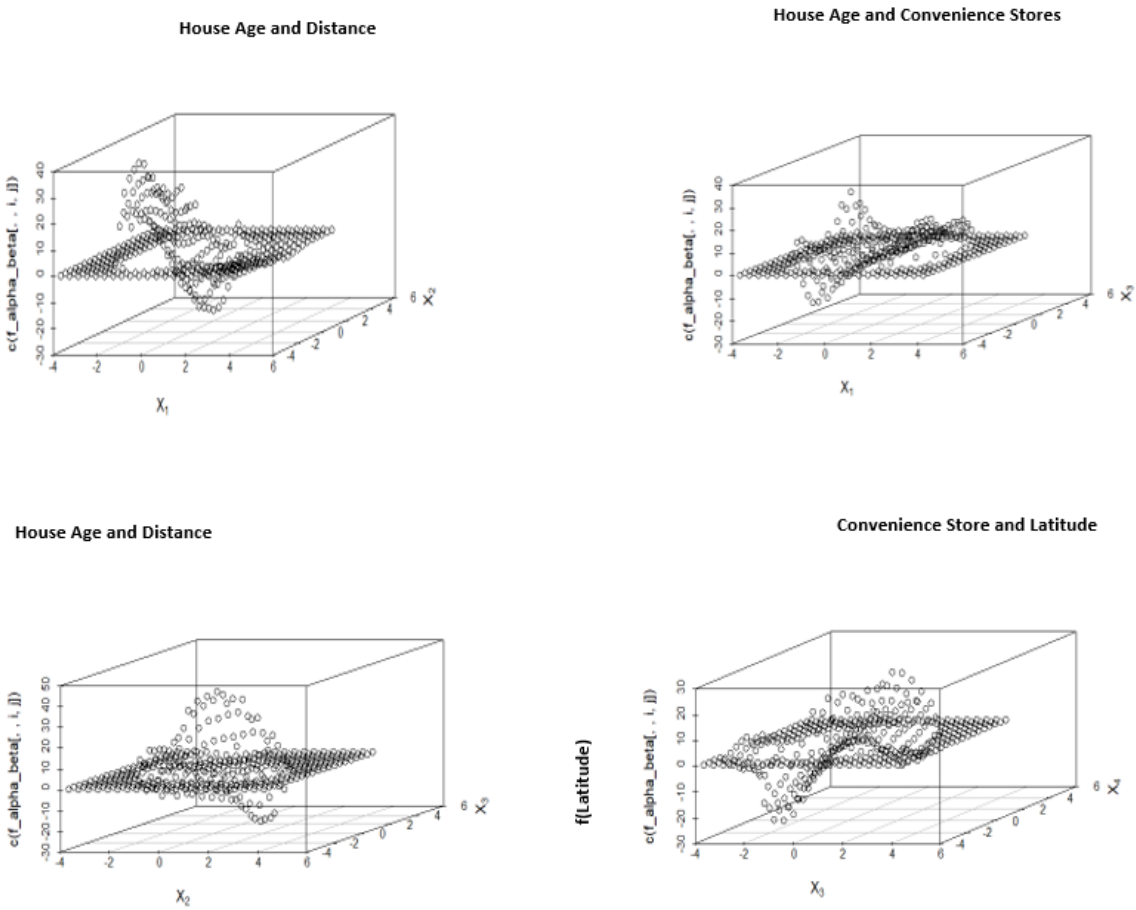

Figure 6.2: Interaction Effects Plots

Figure 4.4 shows the interaction effect plots among the predictors. A second-order

interaction effect is allowed between the predictor variables. The 3D plots of all interactions exhibit non-flatness, which appears contour-like, indicating an interaction between the variables.

6.2 RMPE for Different Methods

In Statistics, Root Mean Prediction Error (RMPE) helps researchers to assess the performance of predictive models and to compare the accuracy of different models. To validate the performance of the method, the RMPE of the Marginal Integration (MI) model is compared with the RMPE of other nonparametric models such as GAM, GAMSEL, Robust Marginal Integration (RMI), and four other linear models including Ordinary Least Regression (OLS), Tukey, Huber, and LASSO penalty. The RMPE of the MI model appears to be the smallest followed by RMI although the difference between the models is not much the MI model appears to perform better than the others. It appears that the RMPE for the nonparametric models performs better than the linear model depicting that the application of the nonparametric model will be a good fit for the data. The results of the RMPE are summarized in the table below.

Table 6.1: RMPE of different estimators for the house price data.

Estimator	MI	GAM	GAMSEL	RMI	OLS	Tukey	Huber	LASSO
RMPE	5.265913	5.54814	5.38276	5.46731	6.66606	6.75857	6.72713	6.66975

Chapter 7

Conclusion

In this thesis, we have studied a Marginal Integration (MI) method for estimating Generalized Additive Model (GAM) with interactions. We compared different nonparametric models (MI, RMI, GAM, GAMSEL) and linear models (OLS, Tukey, Huber, and LASSO) using the root mean prediction error (RMPE). The simulation study indicates that the MI has the least RMPE in pure data when the actual model is nonlinear and has interaction terms. The Robust Marginal Integration (RMI) has the least RMPE when there is a 5% proportion of outliers in the nonlinear model. The RMI method down-weights outliers leading to a relatively small RMPE. The results on RMI are consistent with the method proposed by Boente et al. (2017), indicating robustness in the presence of outliers. Next, we have applied our method to Real Estate Price Prediction data obtained from Kaggle. The nonlinear models performed better than the linear models used to analyze this dataset. The lowest RMPE is produced by the MI method, followed by the RMI.

Bibliography

Bianco, A., Boente, G., González-Manteiga, W. & Pérez-González, A. (2011), 'Asymptotic behavior of robust estimators in partially linear models with missing responses: the effect of estimating the missing probability on the simplified marginal estimators', *Test* **20**(3), 524.

Boente, G., Martínez, A. & Salibián-Barrera, M. (2017), 'Robust estimators for additive models using backfitting', *Journal of Nonparametric Statistics* **29**(4), 744–767.

Chen, R., Härdle, W., Linton, O. B. & Severance-Lossin, E. (1996), Nonparametric estimation of additive separable regression models, *in* 'Statistical Theory and Computational Aspects of Smoothing: Proceedings of the COMPSTAT'94 Satellite Meeting held in Semmering, Austria, 27–28 August 1994', Springer, pp. 247–265.

Cleveland, W. S. (1988), *The Collected Works of John W. Tukey: Graphics 1965-1985*, Vol. 5, CRC Press.

Deaton, A. & Muellbauer, J. (1980), *Economics and consumer behavior*, Cambridge university press.

Fuss, M. (1978), 'A survey of functional forms in the economic analysis of production', *Production economics: A dual approach to theory and application* .

Green, P. J. & Silverman, B. W. (1993), *Nonparametric regression and generalized linear models: a roughness penalty approach*, Crc Press.

Hastie, T. & Tibshirani, R. (1990), 'Generalized additive models london chapman and hall', *Inc* .

Huber, P. J. (1992), 'Robust estimation of a location parameter', *Breakthroughs in statistics: Methodology and distribution* pp. 492–518.

Jiang, Y. (2015), 'Robust estimation in partially linear regression models', *Journal of Applied Statistics* **42**(11), 2497–2508.

Kaggle Real Estate Price Prediction Dataset (n.d.), https://www.kaggle.com/code/mahyamahjoob/real-estate-valuation-using-linear-regression. Accessed: 2023-05-15.

Kong, E., Linton, O. & Xia, Y. (2010), 'Uniform bahadur representation for local polynomial estimates of m-regression and its application to the additive model', *Econometric Theory* **26**(5), 1529–1564.

Leontief, W. (1947), 'Introduction to a theory of the internal structure of functional relationships', *Econometrica, Journal of the Econometric Society* pp. 361–373.

Li, J., Zheng, Z. & Zheng, M. (2012), 'Robust estimation of additive models based on marginal integration'.

Linton, O. & Nielsen, J. P. (1995), 'A kernel method of estimating structured nonparametric regression based on marginal integration', *Biometrika* pp. 93–100.

Mammen, E., Linton, O. & Nielsen, J. (1999), 'The existence and asymptotic properties of a backfitting projection algorithm under weak conditions', *The Annals of Statistics* **27**(5), 1443–1490.

Maronna, R., Martin, R. & Yohai, V. (2006), 'Robust statistics theory and methods john wiley & sons', *Inc., USA* .

Mohd Salleh, R. (2013), A robust estimation method of location and scale with application in monitoring process variability, PhD thesis, Universiti Teknologi Malaysia.

Mosteller, F. & Tukey, J. W. (1977), 'Data analysis and regression. a second course in statistics', *Addison-Wesley series in behavioral science: quantitative methods* .

Ranstam, J. & Cook, J. (2018), 'Lasso regression', *Journal of British Surgery* **105**(10), 1348–1348.

Reid, S., Tibshirani, R. & Friedman, J. (2016), 'A study of error variance estimation in lasso regression', *Statistica Sinica* pp. 35–67.

Rousseeuw, P. J. & Leroy, A. M. (2005), *Robust regression and outlier detection*, John wiley & sons.

Severance-Lossin, E. & Sperlich, S. (1999), 'Estimation of derivatives for additive separable models', *Statistics: A Journal of Theoretical and Applied Statistics* **33**(3), 241–265.

Sperlich, S., Tjøstheim, D. & Yang, L. (2002), 'Nonparametric estimation and testing of interaction in additive models', *Econometric Theory* **18**(2), 197–251.

Stone, C. J. (1985), 'Additive regression and other nonparametric models', *The annals of Statistics* **13**(2), 689–705.

Sun, H. & Liu, Q. (2022), 'Robust empirical likelihood inference for partially linear varying coefficient models with longitudinal data', *Journal of Statistical Computation and Simulation* pp. 1–21.

Tjøstheim, D. & Auestad, B. H. (1994), 'Nonparametric identification of nonlinear time series: projections', *Journal of the American Statistical Association* **89**(428), 1398–1409.

Tukey, J. W. (1977), 'Exploratory data analysis addision-wesley', *Reading, Ma* **688**, 581–582.

Appendix

R CODE

```
Simulation_MI = function(n=50, p=0.05, trim=p, n_test=100, R=100, h_MI=0.9,
h_RMI=0.9,sigma=1, mu_outlier = 20){
  ###### Simulation Code ======= ##

  # n: sample size (training data)
  # p: contamination proportion
  # trim: trimming proportion for RMPE
  # n_test: size of test data
  # R: number of replication
  # sigma: error sd
  # mu_outlier: location of outliers
  set.seed(123)

  library(glmnet) #for LASSO
  library(gamsel)
  library(rmargint)
  library(MASS)
  library(akima) #for interp in MI prediction

  # for gamsel
  n.basis = 5

  # Define the functions and generate the data
```

```
func.v1 <- function(x1) 24*(x1-1/2)^2-2
func.v2 <- function(x2) 2*pi*sin(pi*x2)-4

# X for Training Data
x1 <- runif(n)
x2 <- runif(n)
x3 <- runif(n)
X <- cbind(x1, x2, x3)
reg <- func.v1(x1) + func.v2(x2) + x3 + log(1 + x1*x2) + exp(x2*x3)

# Test data
x1 <- runif(n_test)
x2 <- runif(n_test)
x3 <- runif(n_test)
X_test <- cbind(x1, x2, x3)
reg_test <- func.v1(x1) + func.v2(x2) + x3 + log(1 + x1*x2) + exp(x2*x3)
eps <- rnorm(n_test,0,sd=sigma)
Y_test = reg_test + eps

# Define the bandwidth
bandwd <- matrix(h_RMI, 3, 3)  #Choice of bandwidth for RMI

# prediction errors
RMPE_train_full = RMPE_test = RMPE_train_trim = matrix(NA, R, 8)
colnames(RMPE_train_full) = colnames(RMPE_train_trim) =
  colnames(RMPE_test) = c("RMI", "MI", "OLS", "Huber",
                          "Tukey", "LASSO", "GAM", "GAMSEL")
```

```
for (i in 1:R) {

  if((i==1) | !(i%%1)) cat(sprintf("Step %d/%d\n", i, R))
  eps <- rnorm(n, 0, sd=sigma)
  outlier_index = sample(1:n, round(n*p))
  eps[outlier_index] = rnorm(length(outlier_index),
                      mean = mu_outlier, sd=sigma)

  Y <- reg + eps

  ##### RMI (Robust Marginal Integration) #####
  fit.cl <- margint.rob(Y ~ X, windows=bandwd,type='alpha', degree=1)
  y_pred <- predict(fit.cl, newdata=X)
  # Compute the predicted value in training data
  RMPE_train_trim[i,"RMI"] <- sqrt(upper.trim.mean((y_pred-Y)^2,
                        trim = trim))
  RMPE_train_full[i,"RMI"] <- sqrt(mean((y_pred-Y)^2))

  y_pred <- predict(fit.cl, newdata=X_test)
  # Compute the predicted value in test data
  RMPE_test[i,"RMI"] <- sqrt(mean((y_pred-Y_test)^2))

  ##### MI (Marginal Integration) #### ##
  gam_mi_fit = GAM_MI(Y=Y, X=X, h=h_MI)
  # prediction on training data
  y_pred = predict_MI(Y=Y, X_test=X, x_alpha=gam_mi_fit$x_alpha,
    f_alpha=gam_mi_fit$f_alpha, f_alpha_beta=gam_mi_fit$f_alpha_beta)
```

```
RMPE_train_full[i,"MI"] = sqrt(mean((Y - y_pred)^2))
RMPE_train_trim[i,"MI"] =
  sqrt(upper.trim.mean((Y - y_pred)^2, trim=trim))

# prediction on test data
y_pred = predict_MI(Y=Y, X_test=X_test, x_alpha=gam_mi_fit$x_alpha,
  f_alpha=gam_mi_fit$f_alpha,f_alpha_beta=gam_mi_fit$f_alpha_beta)

RMPE_test[i,"MI"] = sqrt(mean((Y_test - y_pred)^2))

###### OLS ######
lm_fit = lm(Y ~ X)
beta_lm = lm_fit$coefficients #estimator of beta
RMPE_train_full[i,"OLS"] = sqrt(mean((cbind(1, X) %*% beta_lm - Y)^2))
RMPE_train_trim[i,"OLS"] = sqrt(upper.trim.mean((cbind(1, X) %*%
                              beta_lm -Y)^2, trim=trim))
RMPE_test[i,"OLS"] = sqrt(mean((cbind(1, X_test) %*%
                     beta_lm - Y_test)^2))

###### Huber ######
Huber_fit = rlm(Y ~ X, maxit = 100)
beta_Huber = Huber_fit$coefficients #estimator of beta
RMPE_train_full[i,"Huber"] = sqrt(mean((cbind(1, X) %*%
                             beta_Huber - Y)^2))
RMPE_train_trim[i,"Huber"] = sqrt(upper.trim.mean((cbind(1, X) %*%
                           beta_Huber- Y)^2, trim=trim))
RMPE_test[i,"Huber"] = sqrt(mean((cbind(1, X_test) %*%
                                  beta_Huber - Y_test)^2))
```

```
###### Tukey ======##
Tukey_fit = rlm(Y ~ X, method = "MM", maxit = 100)
beta_Tukey = Tukey_fit$coefficients #estimator of beta
RMPE_train_full[i,"Tukey"] = sqrt(mean((cbind(1, X) %*%
                                beta_Tukey - Y)^2))
RMPE_train_trim[i,"Tukey"] = sqrt(upper.trim.mean((cbind(1, X)
      %*% beta_Tukey- Y)^2, trim = trim))
RMPE_test[i,"Tukey"] = sqrt(mean((cbind(1, X_test) %*%
                    beta_Tukey- Y_test)^2))


###### LASSO ======##
lasso_cv = cv.glmnet(x = X, y=Y, alpha = 1, nlambda = 100)
lambda_opt_lasso = lasso_cv$lambda.min #minimum MSE
lasso_fit = glmnet(x = X, y = Y, lambda = lambda_opt_lasso)
#estimators
beta_lasso = as.numeric(coef(lasso_fit))
RMPE_train_full[i,"LASSO"] = sqrt(mean((cbind(1, X) %*%
                                    beta_lasso - Y)^2))
RMPE_train_trim[i,"LASSO"] = sqrt(upper.trim.mean((cbind(1, X)
                %*% beta_lasso - Y)^2, trim=trim))
RMPE_test[i,"LASSO"] = sqrt(mean((cbind(1, X_test)
                        %*% beta_lasso - Y_test)^2))


###### GAM and GAMSEL ======##
n.basis_gamsel = rep(n.basis, ncol(X)) #P number of variables
```

```
n.x_unique = apply(X, 2, function(t) length(unique(t)))
low_degree_index = (n.x_unique <= n.basis)
n.basis_gamsel[low_degree_index] = n.x_unique[low_degree_index] - 1

bases = pseudo.bases(X, degree=n.basis_gamsel, df=4)
gamsel_cv = cv.gamsel(x = X, y = Y, family="gaussian", bases=bases)
lambda_opt_gamsel = gamsel_cv$lambda.min
# gamsel needs at least two values of lambda
temp_lambda_gamsel = c(gamsel_cv$lambda.min, 0)
gamsel_fit = gamsel(x = X, y = Y, lambda = temp_lambda_gamsel,
                    bases=bases, family="gaussian")

###### GAMSEL ======##
# for training data
Y_hat_gamsel = predict(object=gamsel_fit, X)[,1]
RMPE_train_full[i, "GAMSEL"] = sqrt(mean((Y_hat_gamsel - Y)^2))
RMPE_train_trim[i, "GAMSEL"] = sqrt(upper.trim.mean((Y_hat_gamsel
                                    - Y)^2, trim=trim))

#for test data
Y_hat_gamsel = predict(object=gamsel_fit, X_test)[,1]
RMPE_test[i, "GAMSEL"] = sqrt(mean((Y_hat_gamsel - Y_test)^2))



###### MPE for ordinary GAM ======##
# for training data
Y_hat_gam = predict(object=gamsel_fit, X)[,2]
RMPE_train_full[i,"GAM"] = sqrt(mean((Y_hat_gam - Y)^2))
```

```
    RMPE_train_trim[i,"GAM"] = sqrt(upper.trim.mean((Y_hat_gam - Y)^2,
                                    trim=trim))

    #for test data
    Y_hat_gam = predict(object=gamsel_fit, X_test)[,2]
    RMPE_test[i,"GAM"] = sqrt(mean((Y_hat_gam - Y_test)^2))
  }

  average_RMPE_train_full = colMeans(RMPE_train_full)
  average_RMPE_train_trim = colMeans(RMPE_train_trim)
  average_RMPE_test = colMeans(RMPE_test)

  # Print root mean prediction errors over
  cat(sprintf("Root mean prediction errors over %d samples for %g%% outliers
  :\n\n", n, p))
  cat(sprintf("In full training data:\n"))
  print(average_RMPE_train_full)
  cat(sprintf("\n\nIn %g%% trimmed training data:\n", 100*trim))
  print(average_RMPE_train_trim)
  cat(sprintf("\n\nIn test data:\n"))
  print(average_RMPE_test)
  browser()

}

# ------------------------------------------------------------
#     Fitting GAM model with interaction using marginal integration method
# ------------------------------------------------------------
```

```
GAM_MI = function(Y, X, k=21, is.plot=FALSE, h=0.9){
  # Y: (n vector) response variable
  # X: (n*d matrix) covariates
  # k: number of points taken in scaled X
  # is.plot: (logical) if TRUE, plots main and interaction effects

  if(is.plot) library("scatterplot3d")
  n=nrow(X)
  d = ncol(X)
  X = apply(X, 2, scale)#scaling X's


  #Computing M_hat interaction and F hat interaction
  x_alpha = seq(from=min(X), to=max(X), length.out=k)
  h_vec =  rep(h,d  )
   # for (i in 1:d)
   #  h_vec[i]=1.06 * n^(-1/5) * sd(X[, i])


  # plot main effects
  f_alpha = matrix(NA, k, d)
  for (i in 1:d){
    f_alpha[,i] = main_effect(Y=Y, X=X, alpha=i, x_alpha=x_alpha,
                              h_vec=h_vec, n=n, d=d, k=k)
    #plot(x_alpha, f_alpha[,i])
  }

  # interaction effects
```

```
  f_alpha_beta = array(NA, dim = c(k, k, d, d))
  for (i in 1:(d-1))
    for (j in (i+1):d){
      f_alpha_beta[, ,i,j] = interaction_effect(Y=Y, X=X, alpha=i,
                                        beta=j,
                x_alpha=x_alpha,x_beta=x_alpha,
                h_vec=h_vec, n=n, d=d, k=k)
      if (is.plot) scatterplot3d(rep(x_alpha, each=k),
                           rep(x_alpha, k),
      c(f_alpha_beta[, ,i,j]),
    xlab = bquote(X[.(i)]), ylab = bquote(X[.(j)]))
    }

  output = list(x_alpha=x_alpha, f_alpha=f_alpha,
            f_alpha_beta=f_alpha_beta)
  return(output)
}

# ------------------------------------------------------------
#                      main effect
# ------------------------------------------------------------
main_effect = function(Y, X, alpha, x_alpha,
                       h_vec, n, d, k){
  # Y: (n vector) response variable
  # X: (n*d matrix) covariates
  # k: number of points taken in scaled X
  # alpha: (scalar, 1 to d) column index of X
  # x_alpha: (k vector) scaled X space, where f_alpha will be computed
```

```
# h_vec: (d vector) bandwidth parameter

# output: (k*d matrix) estimate of the main
#effect at x_alpha for each column

Y = Y - mean(Y)
F_hat_alpha = rep(NA, k)
m_hat_alpha = rep(NA, n)
j = 1
for (xa in x_alpha){
  Z_alpha = cbind(1, X[, alpha] - xa)

  for (l in 1:n){
    W_l_alpha = kernel_fun(X[, alpha], xa, h_vec[alpha])/n

    for (d1 in 1:d){
      if (d1==alpha) next else
        W_l_alpha = W_l_alpha * kernel_fun(X[, d1],
                        X[l, d1], h_vec[d1])
    }

    if (sum(W_l_alpha) < .0001) {
      m_hat_alpha[l] = 0
      next
    }

    W_l_alpha = diag(W_l_alpha)
    ZW = t(Z_alpha) %*% W_l_alpha
```

```
      m_hat_alpha[l] = c(1, 0) %*% solve(ZW %*% Z_alpha,
                                         ZW %*% Y)
    }

    F_hat_alpha[j] = mean(m_hat_alpha)
    j = j + 1
  }

  return(F_hat_alpha)
}

# ------------------------------------------------------------
#                    interaction effect
# ------------------------------------------------------------
interaction_effect = function(Y, X, alpha, beta, x_alpha, x_beta,
                              h_vec, n, d, k){
  # Y: (n vector) response variable
  # X: (n*d matrix) covariates
  # k: number of points taken in scaled X
  # alpha, beta: (scalar, 1 to d) column index of X
  # x_alpha, x_beta: (k vector) scaled X space, where f_alpha_beta
  #will be computed
  # h_vec: (d vector) bandwidth parameter

  # output: (k*k*d*d array) estimate of the interaction effect at
  #     (x_alpha, x_beta) grid for each pair of columns

  Y = Y - mean(Y)
```

```
F_hat_alpha_beta = matrix(NA, k, k)
m_hat_alpha_beta = rep(NA, n)
s = 1
for (xa in x_alpha){
  t = 1
  for (xb in x_beta){
    Z_alpha_beta = cbind(1, X[, alpha] - xa,  X[, beta] - xb)


    for (l in 1:n){
      W_l_alpha_beta = kernel_fun(X[, alpha], xa, h_vec[alpha]) *
        kernel_fun(X[, beta], xb, h_vec[beta])/n

      for (d1 in 1:d){
        if (d1==alpha | d1==beta) next else
          W_l_alpha_beta = W_l_alpha_beta * kernel_fun(X[, d1],
                                  X[l, d1], h_vec[d1])
      }

      if (sum(W_l_alpha_beta) < 0.0001) {
        m_hat_alpha_beta[l] = 0
        next
      }

      W_l_alpha_beta = diag(W_l_alpha_beta)
      ZW = t(Z_alpha_beta) %*% W_l_alpha_beta
      m_hat_alpha_beta[l] = c(1, 0, 0) %*% solve(ZW
                          %*% Z_alpha_beta, ZW %*% Y)
```

```
      }

      F_hat_alpha_beta[s, t] = mean(m_hat_alpha_beta)
      t = t + 1
    }
    s = s + 1
  }

  return(F_hat_alpha_beta)
}



# ------------------------------------------------------------
#                      Kernel function
# ------------------------------------------------------------
kernel_fun = function(x, mu, h){
  # normal kernel function

  return(dnorm(x, mu, h))
  # x = (x - mu)/h
  # f = (15/16) * (1 - x^2)^2
  # zero_index = (abs(x) > 0)
  # n_zero = sum(zero_index)
  # if (n_zero > 0) f[zero_index] = rep(0, n_zero)
  # return(f)
}

# ------------------------------------------------------------
```

```
#                        Prediction
# ------------------------------------------------------------
predict_MI = function(Y, X_test, x_alpha, f_alpha, f_alpha_beta){
  # Y: (n vector) response variable for training data
  # X: (n*d matrix) covariates for training data
  #(not needed as x_alpha is given)
  # X_test: (n_test*d matrix) test data
  # x_alpha: (k vector) scaled X space, where f_alpha and f_alpha_beta
  #    are computed
  # f_alpha: (k*d matrix) main effect at x_alpha
  # f_alpha_beta: (k*k*d*d array) interaction effect evaluated at a grid
  #      of (x_alpha, x_alpha)

  #library(akima)

  n.test = nrow(X_test)
  d = ncol(X_test)
  k = length(x_alpha)

  #scaling
  X_center = colMeans(X_test)
  X_scale = apply(X_test, 2, sd)
  #X = scale(X,  center=X_center, scale=X_scale) #apply(X, 2, scale)
  X_test = scale(X_test,  center=X_center, scale=X_scale)

  mu = mean(Y)
  y_hat = rep(mu, n.test)
```

```
  # main effects
  for (i in 1:d){
    y_hat = y_hat + approx(x=x_alpha, y=f_alpha[,i],
                           xout = X_test[,i], rule=2)$y
  }

  # interaction effects
  for (i in 1:(d-1))
    for (j in (i+1):d)
      y_hat = y_hat + diag(interp(x=rep(x_alpha, times=k),
                                  y=rep(x_alpha,each=k),
                                    z=c(f_alpha_beta[, ,i,j]),
                                    xo=X_test[,i], yo=X_test[,j],
                                    linear = FALSE, extrap = TRUE)$z)

  return(y_hat)
}

# ------------------------------------------------------------------------
#                  Upper trimmed mean
# ------------------------------------------------------------------------
upper.trim.mean = function(x,trim=0.05) {
  #trim: the fraction of observations to be trimmed from the top
  if (trim==0) return(mean(x))

  x <- sort(x)
  mean(x[1:floor(length(x)*(1-trim))])
}
```

```
# ------------------------------------------------------------------

Simulation_MI(n=50, p=0, h_MI=0.9, h_RMI=0.9, R=100,trim=0.05)
Simulation_MI(n=50,  p=0.05, h_MI=0.9, h_RMI=0.9, R=100,trim=0.05)

Simulation_MI(n=100,  p=0, h_MI=0.7, h_RMI=0.7, R=100,trim=0.05)
Simulation_MI(n=100,  p=0.05, h_MI=0.7, h_RMI=0.7, R=100,trim=0.05)

Simulation_MI(n=200,  p=0, h_MI=0.65, h_RMI=0.65, R=100,trim=0.05)
Simulation_MI(n=200,  p=0.05, h_MI=0.65, h_RMI=0.65, R=100,trim=0.05)
```

Curriculum Vitae

On November 11, 1997, Tahiru Mahama, the last child of Baba Tahiru and Fati Abdulai, was born. After graduating from high school in 2016, he enrolled in Kwame Nkrumah University of Science and Technology (KNUST) in Ghana to pursue a BSc in Statistics. He graduated from the Department of Statistics and Actuarial Science in 2020 as one of the top three students. He held a number of leadership positions at KNUST while completing his degree, including secretary of the Association of Statistics and Actuarial Science Students and vice president of Atmosphere of God's Power. He performed his national service in 2020 as a teaching and research assistant at the KNUST (Ghana) Department of Statistics and Actuarial Science. Tahiru enrolled in graduate school at The University of Texas at EL Paso in Fall 2021 to pursue a Master's in Statistics and Data Science, laying the groundwork for a future PhD in Biostatistics, with the goal of becoming a researcher and professional in the field. He often participated in and contributed to several workshops and seminars on statistics, data science, and mathematics. He held a position as a Teaching and Research Assistant while working toward a Master's degree in Statistics and Data Science. Tahiru plans to pursue a doctorate after graduation in Biostatistics at the School of Public Health, Indiana University, Bloomington.

E-mail address: *tmahama@miners.utep.edu*